AuTom-dualx: a toolkit for fully automatic fiducial marker-based alignment of dual-axis tilt series with simultaneous reconstruction

Renmin Han[1], Xiaohua Wan[2], Lun Li[2,3], Albert Lawrence[4], Peng Yang[1], Yu Li[1], Sheng Wang[1], Fei Sun[5], Zhiyong Liu[2], Xin Gao[1,*], Fa Zhang[2,*]

[1]King Abdullah University of Science and Technology (KAUST), Computational Bioscience Research Center (CBRC), Computer, Electrical and Mathematical Sciences and Engineering (CEMSE) Division, Thuwal, 23955-6900, Saudi Arabia

[2]High Performance Computer Research Center, Institute of Computing Technology, Chinese Academy of Sciences, Beijing, 100190, China

[3]University of Chinese Academy of Sciences, Beijing, China

[4]National Center For Microscopy and Imaging Research, Center for Research in Biological Systems, University of California, 9500 Gilman Drive La Jolla, San Diego, CA, 92093-0608, USA

[5]National Laboratory of Biomacromolecules, Institute of Biophysics, Chinese Academy of Sciences, 15 Datun Road, Beijing 100101, China

[*] All correspondence should be addressed to Xin Gao (xin.gao@kaust.edu.sa) and Fa Zhang (zhangfa@ict.ac.cn).

S1 Technical details

S1.1 Clustering for fiducial marker detection

The fiducial marker detection problem is reduced to a sampling and clustering problem in our procedure. Here, we would like to introduce the details of clustering for fiducial marker detection.

In our previous *Sampling* procedure, the whole micrograph has been divided into several subareas. Within each subarea, a peak position is detected. These peak positions will be considered as the candidates of fiducial marker positions. Each peak has the correlation value as the score (feature) of shape similarity and the pixel value as the score (feature) of contrast similarity. The two scores compose the feature space for the candidates, as shown in Figure S1 (the scores shown here have been normalized and have already been filtered by a gross threshold of 0.5 for shape).

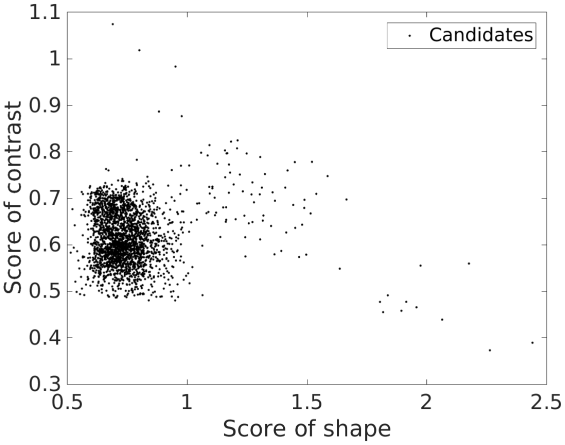


Figure S1: A demonstration of the feature space for the fiducial marker candidates (from the Centriole dataset).

With the feature space, the detected positions can be classified into two clusters, i.e., the fiducial marker cluster and background (non-fiducial marker) cluster. It should be noticed that the fiducial markers always have a higher contrast and better score of shape. We can find that there are thousands of background points and hundreds of fiducial markers. Therefore, it is an unbalanced clustering problem. A density-based clustering (Ankerst *et al.*, 1999) or density estimation is feasible to solve the problem.

Inferred from the central limit theorem, the center of background points follows a Gaussian-like distribution in the feature space, which makes an expectation-maximization (E-M) clustering with Gaussian mixture model possible. Our solution follows the classic Gaussian mixture model, in which the distribution of the two clusters are modeled as Gaussian and the whole system can be written as a linear superposition of Gaussian in the form as

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} p(k)p(x|k) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k), \tag{1}$$

where the kth component is characterized by Gaussian with weight π_k ($\sum_{k=1}^{K} \pi_k = 1$), mean $\boldsymbol{\mu}_k$ and covariance matrix $\boldsymbol{\Sigma}_k$. $K = 2$ and all $\boldsymbol{x}$ are the 2D features of the peak positions. Denoting $\boldsymbol{x} = [x, y]^T$, letting X represent the score of shape and Y represent the score of contrast, we have

$$\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{2\pi\sigma_X\sigma_Y\sqrt{1-\rho^2}} \exp\left(-\frac{1}{2(1-\rho^2)}\left[\frac{(x-\mu_X)^2}{\sigma_X^2} + \frac{(y-\mu_Y)^2}{\sigma_Y^2} - \frac{2\rho(x-\mu_X)(y-\mu_Y)}{\sigma_X\sigma_Y}\right]\right), \tag{2}$$

and

$$\boldsymbol{\mu} = \begin{pmatrix} \mu_X \\ \mu_Y \end{pmatrix}, \quad \boldsymbol{\Sigma} = \begin{pmatrix} \sigma_X^2 & \rho\sigma_X\sigma_Y \\ \rho\sigma_X\sigma_Y & \sigma_Y^2 \end{pmatrix}, \tag{3}$$

where μ_X and μ_Y are the mean values of X and Y, σ_X and σ_Y are the standard deviations of X and Y, and ρ is the correlation between X and Y.

Given the features of selected peaks $\boldsymbol{x}_i, i = 1, .., N$, our goal is to maximize the following objective function:

$$Q = \sum_{i=1}^{N} \log \left(\sum_{k=1}^{2} \pi_k \mathcal{N}(\boldsymbol{x}_i | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \right). \tag{4}$$

The problem here is solved by expectation-maximization algorithm. In the expectation step, we assume all the parameters of the Gaussian mixture are fixed (the values of π_k, $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ are from the initial values or last turn's maximization step) and find the probability $\gamma(i, k)$ for each peak position that belongs to the background cluster ($k = 1$) or fiducial marker cluster ($k = 2$).

E-step:

$$\gamma(i, k) = \frac{\pi_k \mathcal{N}(\boldsymbol{x}_i | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{2} \pi_j \mathcal{N}(\boldsymbol{x}_i | \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}. \tag{5}$$

At the second step (maximization step), the mixture parameters will be further refined.

M-step:

$$\pi_k = \frac{1}{N} \sum_{i=1}^{N} \gamma(i, k), \quad \boldsymbol{\mu}_k = \frac{1}{N\pi_k} \sum_{i=1}^{N} \gamma(i, k) \boldsymbol{x}_i, \quad \boldsymbol{\Sigma}_k = \frac{1}{N\pi_k} \sum_{i=1}^{N} \gamma(i, k)(\boldsymbol{x}_i - \boldsymbol{\mu}_k)(\boldsymbol{x}_i - \boldsymbol{\mu}_k)^T. \tag{6}$$

Repeat the E-step and M-step until convergence. An $\boldsymbol{x}_i$ will be classified to fiducial markers if and only if $\gamma(i, 2) > \gamma(i, 1)$.

To achieve a good estimation, the initial partition of the candidates is also important. Our initial guess of the fiducial markers is based on experiential observations and the following facts: For the normal distribution, one standard deviation above and below the mean encompasses about 68% of the area; and two standard deviations above and below the mean encompasses about 95% of the area. Here, we calculate the XY and the corresponding μ_{XY} and σ_{XY}. For a candidate, if $score_{contrast} \times score_{shape} < \mu_{XY} + \sigma_{XY}$, it is considered as background; and if $score_{contrast} \times score_{shape} > \mu_{XY} + 2\sigma_{XY}$, it is considered as a fiducial marker. Based on this rule, the E-M clustering is initialized. Figure S2(A) shows the initialization of the clustering (data extracted from one micrograph of Centriole), where the yellow ones are the ambiguous candidates. After the convergence, the features from backgrounds and fiducial markers have been clearly classified. By our previous investigation, the E-M clustering has $\geq 90\%$ fiducial marker detection accuracy for normal datasets.

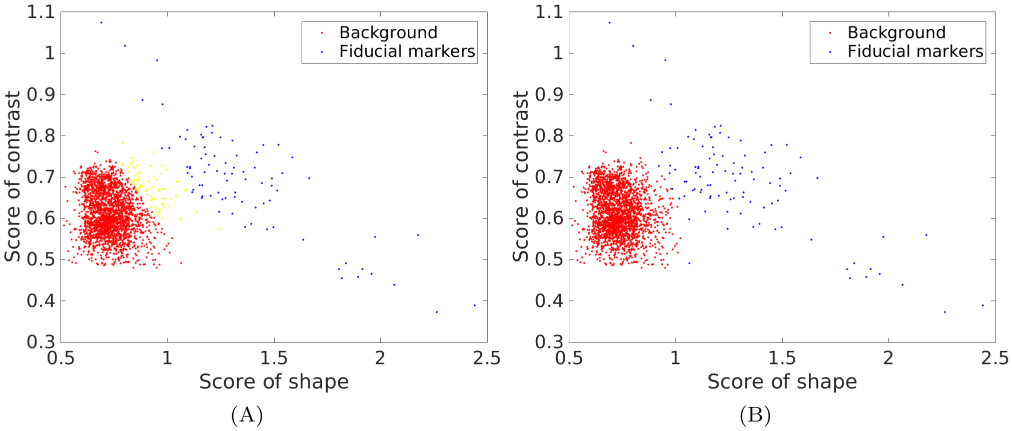


Figure S2: Illustration of the clustering in the fiducial marker detection. (A) The initialization of the E-M clustering. (B) The convergence of the E-M clustering.

S1.2 Fiducial marker tracking

According to the most recent proof of the relationship between the projection model and the tracking model (Han *et al.*, 2018), it is possible to track the fiducial markers only based on their 2-dimensional (2D) projections as the positions of fiducial markers on two micrographs are related by affine transformation within a very small deviation. Here, the fiducial marker tracking problem is reduced to an incomplete point set registration problem:

Let "point set" denote the positions of the fiducial markers extracted from a projection. Given two point sets ($\mathcal{M}$ and $\mathcal{S}$) belonging to different views, an affine transformation $\mathcal{T}(\cdot)$ applied to the moving "model" point set $\mathcal{M}$ can be found so that there exists a subset of $\mathcal{T}(\mathcal{M})$ with the maximum cardinality in which the points are corresponding to the points from a subset of the static "scene" set $\mathcal{S}$ under a selected measure of distance.

Two different tracking algorithms are implemented in our proposal: (i) the RANSAC-based tracking algorithm (Han *et al.*, 2015) that can ensure the robustness to noise with relatively slow execution time, and (ii) the fast tracking algorithm (Han *et al.*, 2018) that solves the tracking problem by algebraic optimization.

S1.2.1 RANSAC-based tracking

In the RANSAC-based tracking algorithm, the point sets from two different projections are divided into "query set" and "reference set". Different query sets are generated and fitted into the reference set with an affine transformation. If the estimated transformation does not degenerate and can cover enough points, the transformation is recorded as a candidate of the global transformation between the query set and reference set. Repeat the procedure until there is no improvement of the mapping residual or it reaches the termination of the consensus query.

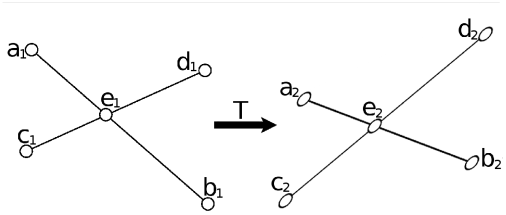


Figure S3: Illustration of a four-point congruent set.

The query tree is built based on the reference of four-point-set invariants. A set of four points (in the following we call it quadrilateral) on one surface consists a system that keeps the cross-ratio invariant under affine transformation, which has been widely used in computer vision (Aiger *et al.*, 2008). Figure S3 shows such an example, in which points $\boldsymbol{a}_1, \boldsymbol{b}_1, \boldsymbol{c}_1, \boldsymbol{d}_1$ compose a point set $\mathcal{S}_1$ and a point set $\mathcal{S}_2 = \{\boldsymbol{a}_2, \boldsymbol{b}_2, \boldsymbol{c}_2, \boldsymbol{d}_2\}$ is generated from $\mathcal{S}_1$ under the affine transformation $\mathcal{T}(\cdot)$ ($\mathcal{T}(\boldsymbol{x}_2; \boldsymbol{A}, \boldsymbol{t}) = \boldsymbol{A}\boldsymbol{x}_1 + \boldsymbol{t}$, where $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are vectors representing points in $\mathbb{R}^2$, $\boldsymbol{A}$ is a 2×2 matrix, and $\boldsymbol{t}$ is the translation). For four-point-set, the ratio $r_1 = \|\boldsymbol{a}_1 - \boldsymbol{e}_1\|/\|\boldsymbol{a}_1 - \boldsymbol{b}_1\|$ and $r_2 = \|\boldsymbol{c}_1 - \boldsymbol{e}_1\|/\|\boldsymbol{c}_1 - \boldsymbol{d}_1\|$ are preserved under affine transformation, i.e., $\|\boldsymbol{a}_2 - \boldsymbol{e}_2\|/\|\boldsymbol{a}_2 - \boldsymbol{b}_2\| = r_1$ and $\|\boldsymbol{c}_2 - \boldsymbol{e}_2\|/\|\boldsymbol{c}_2 - \boldsymbol{d}_2\| = r_2$. r_1 and r_2 are used as the search key to reduce the computational complexity in query. Then, considering the effects of noise, a random sample consensus (RANSAC)(Fischler and Bolles, 1981) procedure is designed for robust estimation of $\mathcal{T}(\cdot)$.

Algorithm 1 elaborates the details of RANSAC-based tracking. Algorithm 1 requires the point set $\mathcal{M}$, $\mathcal{S}$ and a distance threshold d as input and outputs the matched pairs between the point set $\mathcal{M}$ and $\mathcal{S}$, and the implying transformation $\mathcal{T}(\cdot)$. Firstly, from the point set $\mathcal{S}$ a set of quadrilaterals with wide-base[1] is generalized, i.e., $\mathcal{Q} = \{Q_i | Q_i \text{ is a 4 point subset of } \mathcal{S}\}$. A KD-tree (Bentley, 1975) $KD(\mathcal{Q})$ is built based on the quadrilaterals' two invariant ratios r_1 and r_2 (r_1 and r_2 are used as the search keys). We then do the following operations:

1. Select a random quadrilateral P_{ran} from $\mathcal{S}$ and search on the $KD(\mathcal{Q})$ to obtain a subset of $\mathcal{Q}$ (Ψ) whose invariant ratios are very close to P_{ran}.
2. With Ψ and P_{ran}, several possible transformations can be calculated. For each $\mathcal{T}(\cdot)$, apply the $\mathcal{T}(\cdot)$ to $\mathcal{M}$ and count the number of points in $\mathcal{T}(\mathcal{M})$ that is congruent with $\mathcal{S}$ under a loose distance threshold

[1] A quadrilateral with wide-base means that the diagonals of the quadrilateral are with wide-base, judging from the distance of two points: Given the maximum of the distances between two arbitrary points inside a shape d_{max}, if the distance of two points compared with d_{max} is not very small, for example, $> 0.4d_{max}$, the two points are wide-base.

$3 \cdot d$ (denoting the operation to count of congruent points between point set $\mathcal{X}$ and $\mathcal{Y}$ under distance d as $\text{Congr}(\mathcal{X}, \mathcal{Y}, d)$ and defining the corresponding congruent points in $\mathcal{X}$ as $\mathcal{C}(\mathcal{X})$).

3. If the cardinality $\text{Congr}(\mathcal{T}(\mathcal{M}), \mathcal{S}, 3 \cdot d)$ produced in the previous step is large enough, try to re-estimate the transformation $\mathcal{T}(\cdot)$ by $\mathcal{C}(\mathcal{M})$ and $\mathcal{C}(\mathcal{S})$, and recount the congruent subset under a tight distance threshold d.
4. If the cardinality returned from $\text{Congr}(\mathcal{T}(\mathcal{M}), \mathcal{S}, \cdot d)$ is larger than the current maximal record C_{max}, save the current value and refresh the termination condition I.
5. Repeat Steps 1∼4 until termination.

Here, as shown in Algorithm 1, we use a two-stage estimation for the transformation to overcome the noise-sensitivity in the least squares estimation. In our solution, the maximum iteration count is refreshed according to $I = \log(1 - p_s)/\log(1 - p_g^N)$, where p_s is the required success probability, p_g is the probability that the randomly selected point from $\mathcal{M}$ also appears in $\mathcal{S}$, and N is the cardinality of the randomly selected dataset ($N = 4$). The distance threshold d can be set according to the value of the fiducial marker diameter.

```
Algorithm 1: RANSAC-based tracking
Input: $\mathcal{M}$, $\mathcal{S}$, $d$
Procedure FindMatch($\mathcal{M}, \mathcal{S}, d$)
    Generate $\mathcal{Q}$ and create search tree $KD(\mathcal{Q})$;
    Calculate maximum iteration count $I$;
    Let $i \leftarrow 0$, $C_{max} \leftarrow 0$;
    while $i < I$ do
        Select a random quadrilateral $P_{ran}$ from $\mathcal{M}$;
        Obtain $\Psi = \{Q_i | Q_i \in KD(\mathcal{Q})\}$ whose keys are close to $P_{ran}$;
        foreach $Q_i \in \Psi$ do
            Estimate $\mathcal{T}(\cdot)$ from $P_{ran}$ to $Q_i$;
            Apply $\mathcal{T}(\cdot)$ to $\mathcal{M}$;
            $c \leftarrow \text{Congr}(\mathcal{T}(\mathcal{M}), \mathcal{S}, 3 \cdot d)$, and obtain $\mathcal{C}(\mathcal{M})$ and $\mathcal{C}(\mathcal{S})$;
            if $c > 0.5 \cdot C_{max}$ then
                Re-estimate $\mathcal{T}(\cdot)$ by $\mathcal{C}(\mathcal{M})$ and $\mathcal{C}(\mathcal{S})$;
                Apply $\mathcal{T}(\cdot)$ to $\mathcal{M}$;
                $c \leftarrow \text{Congr}(\mathcal{T}(\mathcal{M}), \mathcal{S}, d)$;
            end
            if $c > C_{max}$ then
                $C_{max} \leftarrow c$;
                Refresh the maximum iteration count $I$;
            end
        end
        $i \leftarrow i + 1$;
    end
    Return matched pairs between $\mathcal{C}(\mathcal{M})$ and $\mathcal{C}(\mathcal{S})$.
```

S1.2.2 Probability-based fast tracking

Though the RANSAC-based method is robust, it suffers from the issue of increasing execution time when the number of fiducial markers becomes large (> 150). A probability-based fast solution is proposed to compensate the problem of long execution time (Han *et al.*, 2018). In the fast tracking algorithm, the point sets are represented by the probability density function of a Gaussian kernel, i.e., the Gaussian-mixed model (GMM). Here, we first introduce the concept of GMM presentation and the algebraic optimization for ideal conditions of point set registration by coherent point drift (CPD) technique (Myronenko and Song, 2010), and then our overall fast tracking algorithm.

For the convenience of discussion, we explicate the "model" point set as $\mathcal{M} = \{\boldsymbol{y}_m, m = 1, ..., M\}$ and the "scene" set as $\mathcal{S} = \{\boldsymbol{x}_n, n = 1, ..., N\}$.

GMM presentation of a point set: a Gaussian kernel-based probability density function can depict the possibility that a point $\boldsymbol{x}$ is corresponding to another point $\boldsymbol{y}$:

$$p(\boldsymbol{x}|\boldsymbol{y}) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{\|\boldsymbol{x}-\boldsymbol{y}\|^2}{2\sigma^2}\right), \tag{7}$$

where σ is a similarity punishment. It can be noticed that the probability reaches its maximum if the point $\boldsymbol{x}$ is identical to $\boldsymbol{y}$. Similarly, for the point set $\mathcal{M}$, its Gaussian mixture model presentation is defined as $p(\boldsymbol{x}|\mathcal{M}) = \sum_{j=1}^{m} P(m)p(\boldsymbol{x}|\boldsymbol{y}_m)$, where $P(m)$ is the prior probability of the mth point $\boldsymbol{y}_m$. The consequent set to set case ($\mathcal{M}$ to $\mathcal{S}$) can be reduced from the one point to set case as $\frac{1}{2\pi\sigma^2}\sum_{n=1}^{N}\sum_{m=1}^{M} w_m e^{-\frac{\|\boldsymbol{x}_n-\mathcal{T}(\boldsymbol{y}_m)\|^2}{2\sigma^2}}$, where w_m is the weight specified by the prior.

It is clear that the sum of the probability density function has the robust similarity form of $sim_{robust} = \sum\sum g(\|\boldsymbol{x}-\boldsymbol{y}\|^2)$, where $g(\cdot)$ is a Gaussian function. However, it is not a convex function (Jian and Vemuri, 2011). A heuristic solution such as simulated annealing may overcome some local maximums, but it cannot guarantee a global optimization.

Because the missed or added markers caused by noise or tilt change may easily change the distribution of fiducial markers, the similarity punishment is a key to ensure a global similarity. We borrow the concept of the point-spread function (PSF) and full width at half maximum (FWHM) to explain the criteria in σ determination, considering the similarity between the Gaussian kernel and PSF. A σ in PSF defines the closeness of two separate emitter (Betzig *et al.*, 2006; Nieuwenhuizen *et al.*, 2013). Here, we would like to guarantee that almost all the fiducial markers are distinguishable under a given σ, which means that the peak of the Gaussian function should be retained under summation. Therefore, σ can be determined as

$$\sigma = \beta \min\left(\frac{1}{M}\sum_{m=1}^{M} dist(\boldsymbol{y}_m, \mathcal{M}), \frac{1}{N}\sum_{n=1}^{N} dist(\boldsymbol{x}_n, \mathcal{S})\right), \tag{8}$$

where $\beta \in [0.2, 0.5]$ is the parameter to control the distance interpretation and $dist(\boldsymbol{x}, \mathcal{S})$ refers to the minimum distance between the point $\boldsymbol{x}$ to the points belonging to $\mathcal{S}$ (except itself).

Fast parameter refinement by CPD: To optimize the transformation $\mathcal{T}(\cdot)$ in the set to set GMM presentation, coherent point drift (CPD) (Myronenko and Song, 2010) is introduced. The view of CPD is from the aspect of assignment and CPD introduces a representation of the outliers.

Given a point $\boldsymbol{x}$, its probability corresponding to the mth point $\boldsymbol{y}_m$ is $P(m)$, and its total probability that belongs to the point set $\mathcal{M}$ is $\sum_{j=1}^{m} P(m)p(\boldsymbol{x}|\boldsymbol{y}_m)$. Considering the probability that point $\boldsymbol{x}$ may represent an outlier, let w denote the outlier probability. Assuming $\boldsymbol{x}$ is sampled from the point set $\mathcal{S}$ randomly and the points in $\mathcal{M}$ share an equal prior probability, the mixture model takes the form as:

$$p(\boldsymbol{x}) = w\frac{1}{N} + (1-w)\sum_{j=1}^{m}\frac{1}{M}p(\boldsymbol{x}|\boldsymbol{y}_m). \tag{9}$$

The objective function in the negative log-likelihood fashion is

$$E(\mathcal{T}(\cdot), \sigma^2) = -\sum_{n=1}^{N}\log\sum_{m=1}^{M+1} P(m)p(\boldsymbol{x}|\boldsymbol{y}_m), \tag{10}$$

where $P(m)$ here is the reweighted prior and $p(\boldsymbol{x}|\boldsymbol{y}_{M+1}) = \frac{w}{N}$ represents the probability of outliers. The negative likelihood function has an obvious advantage that it can be effectively solved by an expectation-maximization algorithm.

According to Jensen's inequality, the upper bound of the negative log-likelihood function used in the E-M algorithm can be defined as:

$$Q = -\sum_{n=1}^{N}\sum_{m=1}^{M+1} p^{old}(m|\boldsymbol{x}_n)\log\left(P^{new}(m)p^{new}(\boldsymbol{x}_n|\boldsymbol{y}_m)\right), \tag{11}$$

where $p(m|\boldsymbol{x}_n) = P(m)p(\boldsymbol{x}_n|\boldsymbol{y}_m)/p(\boldsymbol{x}_n)$ is the corresponding probability between $\boldsymbol{y}_m$ and $\boldsymbol{x}_n$, the "old" parameters indicate the guess of values in the E-step and the "new" parameters are determined by minimizing the expectation of the complete negative log-likelihood function in the M-step. The algorithm iterates the E-step and M-step until convergence.

E-step:

Ignoring the constants, the objective function in the E-step can be rewritten as:

$$Q(\boldsymbol{A},\boldsymbol{t},\sigma^2)=\frac{1}{2\sigma^2}\sum_{n=1}^{N}\sum_{m=1}^{M}p^{old}(m|\boldsymbol{x}_n)\|\boldsymbol{x}_n-\mathcal{T}(\boldsymbol{y}_m;\boldsymbol{A},\boldsymbol{t})\|^2+\frac{N_pD}{2}log\sigma^2, \tag{12}$$

where $\mathcal{T}(\boldsymbol{y}_m;\boldsymbol{A},\boldsymbol{t})=\boldsymbol{A}\boldsymbol{y}_m+\boldsymbol{t}$, $N_p=\sum_{n=1}^{N}\sum_{m=1}^{M}p^{old}(m|\boldsymbol{x}_n)\leqslant N$ (with $N=N_p$ only if $w=0$), and $p^{old}(m|\boldsymbol{x}_n)$ denotes the posterior probabilities of GMM components calculated using the previous parameter values:

$$p^{old}(m|\boldsymbol{x}_n)=\frac{\exp\left(-\frac{1}{2}\|\frac{\boldsymbol{x}_n-\mathcal{T}(\boldsymbol{y}_m,\boldsymbol{A}^{old},\boldsymbol{t}^{old})}{\sigma^{old}}\|^2\right)}{\sum_{k=1}^{M}\exp\left(-\frac{1}{2}\|\frac{\boldsymbol{x}_n-\mathcal{T}(\boldsymbol{y}_k,\boldsymbol{A}^{old},\boldsymbol{t}^{old})}{\sigma^{old}}\|^2\right)+2\pi\sigma^2\frac{w}{1-w}\frac{M}{N}}. \tag{13}$$

Within a feasible local convex, minimizing the objective function Q in the E-step is equal to solving the negative log-likelihood function E in Eq.(10).

M-step:

In the M-step, our aim is to find the optimal $\boldsymbol{A}$ and $\boldsymbol{t}$ that minimize the objective function Q. It is clear that Q has a closed-form expression (Myronenko and Song, 2010). For the convenience of the following discussion, several representations of variables are introduced:

1. $\boldsymbol{X}_{N\times 2}=(\boldsymbol{x}_1\cdots\boldsymbol{x}_N)^T$—matrix presentation of the point set $\mathcal{S}$;
2. $\boldsymbol{Y}_{M\times 2}=(\boldsymbol{y}_1\cdots\boldsymbol{y}_M)^T$—matrix presentation of the point set $\mathcal{M}$;
3. $\mathbf{1}$—the column vector of all ones;
4. $\mathrm{d}(\boldsymbol{a})$—the diagonal matrix formed from vector $\boldsymbol{a}$;
5. $\boldsymbol{P}$—the matrix that is composed by $p_{mn}=p^{old}(m|\boldsymbol{x}_n)$.

The solution is to firstly find the root of the partial derivative Q with respect to $\boldsymbol{t}$ and then substitute $\boldsymbol{t}$ back into the objective function to obtain the matrix $\boldsymbol{A}$. Here the equation of $\frac{\partial Q}{\partial \boldsymbol{t}}$ takes the form as

$$\frac{\partial Q}{\partial \boldsymbol{t}}=\frac{1}{\sigma^2}\sum_{n=1}^{N}\sum_{m=1}^{M}p^{old}(m|\boldsymbol{x}_n)\left(\boldsymbol{x}_n-\boldsymbol{A}\boldsymbol{y}_m+\boldsymbol{t}\right)=0, \tag{14}$$

in which the root $\boldsymbol{t}$ is

$$\boldsymbol{t}\quad=\quad\frac{1}{N_p}\sum_{n=1}^{N}\sum_{m=1}^{M}p^{old}(m|\boldsymbol{x}_n)(\boldsymbol{x}_n-\boldsymbol{A}\boldsymbol{y}_m). \tag{15}$$

By introducing the variables $\boldsymbol{\mu}_x=\frac{1}{N_p}\boldsymbol{X}^T\boldsymbol{P}^T\mathbf{1}$, $\boldsymbol{\mu}_y=\frac{1}{N_p}\boldsymbol{Y}^T\boldsymbol{P}\mathbf{1}$, $\boldsymbol{t}$ can be rewritten as $\boldsymbol{t}=\boldsymbol{\mu}_x-\boldsymbol{A}\boldsymbol{\mu}_y$.

Substitute $\boldsymbol{t}$ back into Q and solve the equation of partial derivative $\frac{\partial Q}{\partial \boldsymbol{A}}$, the root $\boldsymbol{A}$ is

$$\boldsymbol{A}=(\hat{\boldsymbol{X}}^T\boldsymbol{P}^T\hat{\boldsymbol{Y}})(\hat{\boldsymbol{Y}}^T\mathrm{d}(\boldsymbol{P}\mathbf{1})\hat{\boldsymbol{Y}})^{-1}, \tag{16}$$

where $\hat{\boldsymbol{X}}=\boldsymbol{X}-\mathbf{1}\boldsymbol{\mu}_x^T$, $\hat{\boldsymbol{Y}}=\boldsymbol{Y}-\mathbf{1}\boldsymbol{\mu}_y^T$. σ^2 is also updated according to the result of $\frac{\partial Q}{\partial \sigma^2}$:

$$\sigma^2=\frac{1}{N_pD}(\mathrm{tr}(\hat{\boldsymbol{X}}^T\mathrm{d}(\boldsymbol{P}^T\mathbf{1})\hat{\boldsymbol{X}})-\mathrm{tr}(\hat{\boldsymbol{X}}^T\boldsymbol{P}^T\hat{\boldsymbol{Y}}\boldsymbol{A}^T)). \tag{17}$$

After solving the matrix $\boldsymbol{A}$ and $\boldsymbol{t}$, the moving "model" point set $\mathcal{M}$ is then updated by $\mathcal{T}(\boldsymbol{y}_m)=\boldsymbol{A}\boldsymbol{y}_m+\boldsymbol{t}$.

Overall fast tracking algorithm: As an algebraic solution for non-convex function, CPD still has the problem of local convergence. Our method tries to overcome the local convergence issue. Given two sets of fiducial markers $\mathcal{S}$ and $\mathcal{M}$, Algorithm 2 illustrates the procedure of fiducial marker tracking based on the Gaussian mixture model.

The initial value of σ is assigned according to Eq.(8). As a compensation, a grid search of the applicable value of the transformation matrix $\boldsymbol{A}$ and $\boldsymbol{t}$ with a step size of 2.5σ is conducted at the beginning. The initial values of $\boldsymbol{A}$ and $\boldsymbol{t}$ with a high value of the robust similarity measure is fed into CPD. Under the assumption that the pitch angle almost remains unchanged, we can get the simplified version of $\boldsymbol{A}$ according to the derivation in Han *et al.* (2018):

$$\boldsymbol{A}=\frac{s_{i'}}{s_i}\boldsymbol{R}_{\gamma_{i'}}\begin{pmatrix}\frac{\cos\beta_{i'}}{\cos\beta_i} & \frac{\tan\alpha\sin\Delta\beta}{\cos\beta_i}\\ 0 & 1\end{pmatrix}\boldsymbol{R}_{-\gamma_i}. \tag{18}$$

Algorithm 2: Probability-based fast tracking

```
Input: $\mathcal{M}$,$\mathcal{S}$
Procedure FindMatch($\mathcal{M}, \mathcal{S}$)
    Generate initial estimation of $\mathcal{T}(\cdot; \boldsymbol{A}, \boldsymbol{t})$ by grid search;
    Select several $\{\mathcal{T}(\cdot)\}$ as initial inputs;
    foreach $\mathcal{T}_i(\cdot) \in \{\mathcal{T}(\cdot)\}$ do
        Estimate $\mathcal{T}_i(\cdot)$ by CPD as in previous paragraph;
        Apply $\mathcal{T}_i(\cdot)$ to $\mathcal{M}$;
        Calculate $sim_{robust}(\mathcal{S}, \mathcal{T}_i(\mathcal{M}))$;
        if $sim_{robust}$ is large enough then
            Use $\mathcal{T}_i(\cdot)$ and stop trying;
        end
    end
    $\mathcal{T}(\cdot) \leftarrow \mathcal{T}_i(\cdot)$;
    Return $\mathcal{T}(\cdot)$.
```

where $\beta_{i'}$ and β_i are the corresponding tilt angles of the two micrographs, $\Delta\beta = \beta_{i'} - \beta_i$, $\gamma_{i'}$ and γ_i are the in-plane rotation, $s_{i'}$ and s_i are the scale change, and α is the patch angle. It should be noted that the values of $s_{i'}$ and s_i are very close to each other. The values of $\beta_{i'}$ and β_i can be assigned from the recorded tilt angles when designing the search range of $\boldsymbol{A}$. Considering the real pitch angle in a tilt series, $\tan\alpha$ is a very small value and can be neglected. Therefore, the search range of $\boldsymbol{A}$ is very small. If the tilt angle interval is not too large, the affine matrix $\boldsymbol{A}$ will degenerate to a rotation matrix. However, it is enough for an initial estimation of $\boldsymbol{A}$. The search range of $\boldsymbol{t}$ is limited in half the width of the micrographs, which is enough considering the actual shift. The transformation which has a high rank of $sim_{robust}(\mathcal{X}, \mathcal{T}_i(\mathcal{Y}))$ will be used as the initial seed for further estimation. The fine estimation of transformation $\mathcal{T}(\cdot)$ is then carried out by CPD. The refined value $\mathcal{T}(\cdot)$ with the highest $sim_{robust}(\mathcal{X}, \mathcal{T}_i(\mathcal{Y}))$ is outputted as the final result.

S1.2.3 Divide-and-conquer strategy to refine the tracking

The affine transformation relationship cannot cover all the correspondence of two sets of fiducial markers under the effect of lens distortion. Fortunately, we notice that the divide-and-conquer strategy can solve this issue well. Our proposed algorithm against the lens distortion is illustrated in Algorithm 3.

Algorithm 3: Refine transformation $\mathcal{T}(\cdot)$ against lens distortion

```
Input: $\mathcal{M}$, $\mathcal{S}$, $\mathcal{T}$, and distance threshold $d$
Procedure RefineTransform($\mathcal{M}, \mathcal{S}, \mathcal{T}, d$)
    $\mathcal{C} \leftarrow \emptyset$; $\Phi \leftarrow \emptyset$;
    while $dist(\mathcal{S}, \mathcal{T}(\mathcal{M})) < d$ do
        Apply $\mathcal{T}(\cdot)$ to $\mathcal{M}$;
        foreach $x \in \mathcal{S}$ and $y \in \mathcal{M}$ do
            if $dist(x, \mathcal{T}(y)) < 0.75d$ then
                $\mathcal{C} \leftarrow \mathcal{C} \cup (x, y)$;
            end
            $\mathcal{C}' \leftarrow \emptyset$;
            if $dist(x, \mathcal{T}(y)) < d$ then
                $\mathcal{C}' \leftarrow \mathcal{C}' \cup (x, y)$;
            end
        end
        $\Phi \leftarrow \Phi \cup \mathcal{T}$;
        Recalculate transformation $\mathcal{T}$ by $\mathcal{C}'$;
    end
    Return $\mathcal{C}$ and $\Phi$.
```

The distance threshold d can be set to the diameter of fiducial markers, the same as in RANSAC-based tracking. The input is the result from RANSAC-based tracking or fast tracking. Firstly, the transformation $\mathcal{T}(\cdot)$ is applied to $\mathcal{M}$ and then the peer distance with $\mathcal{S}$ is calculated. If the distance of two points from different point sets is less than the distance threshold d, the point pair is considered as a matched pair. If the distance is less then $0.75d$, the matched pair will be directly outputted; if not, the matched pair will be used to recalculate the transformation $\mathcal{T}(\cdot)$. Here we assume that the distortion is mild and continuous, as we present in the main manuscript. In the next iteration, we rerun the procedure to generate the satisfied correspondence until no matched point pair can be found between $\mathcal{S}$ and $\mathcal{T}(\mathcal{M})$. Finally, multiple transformation $\mathcal{T}$ will be outputted to describe the distortion in the micrographs.

S1.3 Spatial relationship calibration

For the convenience of discussion, we define the coordinate system of tilt series A as the global coordinate system. Given a rigid transformation that makes $\boldsymbol{r}^B = \boldsymbol{R}_{rigid}\boldsymbol{r}^{A'} + \boldsymbol{t}_{rigid}$, where $\boldsymbol{R}_{rigid}$ and $\boldsymbol{t}_{rigid}$ are the approximate estimation of rigid transformation from $\boldsymbol{r}^A$ to $\boldsymbol{r}^B$, and $\boldsymbol{r}^{A'}$ is the inversion estimation of $\boldsymbol{r}^A$ from $\mathcal{T}^{-1}(\boldsymbol{r}^B)$, the projection model of axis B in the coordinate system of A can be denoted as

$$\begin{pmatrix} u_B \\ v_B \end{pmatrix} = s_B \boldsymbol{R}_{\gamma_B} \boldsymbol{P} \boldsymbol{R}_{\beta_B} \boldsymbol{R}_{\alpha_B} \left(\boldsymbol{R}_{rigid}\boldsymbol{r}^{A'} + \boldsymbol{t}_{rigid} \right) + \boldsymbol{t}_B. \tag{19}$$

Considering the observation of projection $(u_B, v_B)^T$ on the coordinate system of A

$$\begin{pmatrix} u_B \\ v_B \end{pmatrix} = s_{A'} \boldsymbol{R}_{\gamma_{A'}} \boldsymbol{P} \boldsymbol{R}_{\beta_{A'}} \boldsymbol{R}_{\alpha_{A'}} \boldsymbol{r}^{A'} + \boldsymbol{t}_{A'}, \tag{20}$$

by substituting Eq.19 to Eq.20, we can get the equation

$$s_B \boldsymbol{R}_{\gamma_B} \boldsymbol{P} \boldsymbol{R}_{\beta_B} \boldsymbol{R}_{\alpha_B} \left(\boldsymbol{R}_{rigid}\boldsymbol{r}^{A'} + \boldsymbol{t}_{rigid} \right) + \boldsymbol{t}_B = s_{A'} \boldsymbol{R}_{\gamma_{A'}} \boldsymbol{P} \boldsymbol{R}_{\beta_{A'}} \boldsymbol{R}_{\alpha_{A'}} \boldsymbol{r}^{A'} + \boldsymbol{t}_{A'}, \tag{21}$$

where the global parameters $s_{A'}, \gamma_{A'}, \beta_{A'}, \alpha_{A'}, \boldsymbol{t}_{A'}$ are unknown. However, the global translation can be directly calculated by $\boldsymbol{t}_{A'} = s_B \boldsymbol{R}_{\gamma_B} \boldsymbol{P} \boldsymbol{R}_{\beta_B} \boldsymbol{R}_{\alpha_B} \boldsymbol{t}_{rigid} + \boldsymbol{t}_B$. Our following focus is to solve $s_{A'}, \gamma_{A'}, \beta_{A'}, \alpha_{A'}$.

In detail, we have $\boldsymbol{R}_{\alpha_B} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha_B & \sin\alpha_B \\ 0 & -\sin\alpha_B & \cos\alpha_B \end{pmatrix}$, $\boldsymbol{R}_{\beta_B} = \begin{pmatrix} \cos\beta_B & 0 & -\sin\beta_B \\ 0 & 1 & 0 \\ \sin\beta_B & 0 & \cos\beta_B \end{pmatrix}$, $\boldsymbol{P} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$, $\boldsymbol{R}_{\gamma_B} = \begin{pmatrix} \cos\gamma_B & \sin\gamma_B \\ -\sin\gamma_B & \cos\gamma_B \end{pmatrix}$ and $\boldsymbol{R}_{rigid} = \begin{pmatrix} r_0 & r_1 & r_2 \\ r_3 & r_4 & r_5 \\ r_6 & r_7 & r_8 \end{pmatrix}$. By substituting the matrices to Eq.21 and focusing on the parameters of $\boldsymbol{r}^{A'}$, we can get the equation as

$$\begin{aligned} s_{A'} \boldsymbol{R}_{\gamma_{A'}} \quad \boldsymbol{P} \quad \boldsymbol{R}_{\beta_{A'}} \boldsymbol{R}_{\alpha_{A'}} &= s_B \boldsymbol{R}_{\gamma_B} \boldsymbol{P} \boldsymbol{R}_{\beta_B} \boldsymbol{R}_{\alpha_B} \boldsymbol{R}_{rigid} \\ &= \begin{pmatrix} \cos\gamma_B & \sin\gamma_B \\ -\sin\gamma_B & \cos\gamma_B \end{pmatrix} \boldsymbol{P} \begin{pmatrix} \cos\beta_B & \sin\alpha_B\sin\beta_B & -\cos\alpha_B\sin\beta_B \\ 0 & \cos\alpha_B & \sin\alpha_B \\ \sin\beta_B & -\sin\alpha_B\cos\beta_B & \cos\alpha_B\cos\beta_B \end{pmatrix} \begin{pmatrix} r_0 & r_1 & r_2 \\ r_3 & r_4 & r_5 \\ r_6 & r_7 & r_8 \end{pmatrix} \\ &= \begin{pmatrix} p_0 & p_1 & p_2 \\ p_3 & p_4 & p_5 \end{pmatrix}, \end{aligned} \tag{22}$$

where

$$\begin{aligned} p_0 &= s_B\cos\gamma_B(r_0\cos\beta_B - r_6\cos\alpha_B\sin\beta_B + r_3\sin\alpha_B\sin\beta_B) + s_B\sin\gamma_B(r_3\cos\alpha_B + r_6\sin\alpha_B), \\ p_1 &= s_B\cos\gamma_B(r_1\cos\beta_B - r_7\cos\alpha_B\sin\beta_B + r_4\sin\alpha_B\sin\beta_B) + s_B\sin\gamma_B(r_4\cos\alpha_B + r_7\sin\alpha_B), \\ p_2 &= s_B\cos\gamma_B(r_2\cos\beta_B - r_8\cos\alpha_B\sin\beta_B + r_5\sin\alpha_B\sin\beta_B) + s_B\sin\gamma_B(r_5\cos\alpha_B + r_8\sin\alpha_B), \\ p_3 &= s_B\cos\gamma_B(r_3\cos\alpha_B + r_6\sin\alpha_B) - s_B\sin\gamma_B(r_0\cos\beta_B - r_6\cos\alpha_B\sin\beta_B + r_3\sin\alpha_B\sin\beta_B), \\ p_4 &= s_B\cos\gamma_B(r_4\cos\alpha_B + r_7\sin\alpha_B) - s_B\sin\gamma_B(r_1\cos\beta_B - r_7\cos\alpha_B\sin\beta_B + r_4\sin\alpha_B\sin\beta_B), \\ p_5 &= s_B\cos\gamma_B(r_5\cos\alpha_B + r_8\sin\alpha_B) - s_B\sin\gamma_B(r_2\cos\beta_B - r_8\cos\alpha_B\sin\beta_B + r_5\sin\alpha_B\sin\beta_B). \end{aligned} \tag{23}$$

Here, it should be noted that p_0 to p_5 are all constants.

Similarly, we have the details that $\boldsymbol{R}_{\alpha_{A'}} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha_{A'} & \sin\alpha_{A'} \\ 0 & -\sin\alpha_{A'} & \cos\alpha_{A'} \end{pmatrix}$, $\boldsymbol{R}_{\beta_{A'}} = \begin{pmatrix} \cos\beta_{A'} & 0 & -\sin\beta_{A'} \\ 0 & 1 & 0 \\ \sin\beta_{A'} & 0 & \cos\beta_{A'} \end{pmatrix}$, and $\boldsymbol{R}_{\gamma_{A'}} = \begin{pmatrix} \cos\gamma_{A'} & \sin\gamma_{A'} \\ -\sin\gamma_{A'} & \cos\gamma_{A'} \end{pmatrix}$. By substituting the matrices to Eq.23, we can get the following equation:

$$\begin{cases} s_{A'}\cos\beta_{A'}\cos\gamma_{A'} = p_0, \\ s_{A'}(\cos\alpha_{A'}\sin\gamma_{A'} + \cos\gamma_{A'}\sin\alpha_{A'}\sin\beta_{A'}) = p_1, \\ s_{A'}(\sin\alpha_{A'}\sin\gamma_{A'} - \cos\alpha_{A'}\cos\gamma_{A'}\sin\beta_{A'}) = p_2, \\ s_{A'}(-\cos\beta_{A'}\sin\gamma_{A'}) = p_3, \\ s_{A'}(\cos\alpha_{A'}\cos\gamma_{A'} - \sin\alpha_{A'}\sin\beta_{A'}\sin\gamma_{A'}) = p_4, \\ s_{A'}(\cos\gamma_{A'}\sin\alpha_{A'} + \cos\alpha_{A'}\sin\beta_{A'}\sin\gamma_{A'}) = p_5. \end{cases} \tag{24}$$

The estimation of $s_{A'}, \gamma_{A'}, \beta_{A'}, \alpha_{A'}$ in Eq. 24 is a non-linear least squares problem, i.e.,

$$\operatorname{argmin} \sum_{i=0}^{5} \left[f_i(s_{A'}, \gamma_{A'}, \beta_{A'}, \alpha_{A'}) - p_i \right]^2, \tag{25}$$

where f_i represents the left part of the equation. It can be solved by the Gauss-Newton (G-N) algorithm or the Levenberg-Marquardt (L-M) algorithm (Marquardt, 1963). For either algorithm, the Jacobian matrix is necessary. Because the system is composed by 6 equations with 4 parameters, the Jacobian matrix is with 6×4 size, which can be directly derived from the $\frac{\partial f_i}{\partial s_{A'}}$, $\frac{\partial f_i}{\partial \gamma_{A'}}$, $\frac{\partial f_i}{\partial \beta_{A'}}$ and $\frac{\partial f_i}{\partial \alpha_{A'}}$ $(i = 0, 1..., 5)$, i.e,

$$\mathbf{J} = \begin{pmatrix} \dfrac{\partial \mathbf{f}}{\partial s_{A'}} & \cdots & \dfrac{\partial \mathbf{f}}{\partial \alpha_{A'}} \end{pmatrix} = \begin{pmatrix} \dfrac{\partial f_0}{\partial s_{A'}} & \cdots & \dfrac{\partial f_0}{\partial \alpha_{A'}} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial f_5}{\partial s_{A'}} & \cdots & \dfrac{\partial f_5}{\partial \alpha_{A'}} \end{pmatrix}. \tag{26}$$

For the tilt series B, there are multiple micrographs. We can either solve the initial transformation on each individual micrograph or with the whole tilt series. However, from the derivation of Eq. 24 and Eq. 26, we can find that the parameters of each individual micrograph have no relationship with each other, which results in the zero value of Jacobian attributes. From the G-N algorithm and L-M algorithm, we can find that zero values of Jacobian attributes have no contribution to the convergence. Therefore, either solving the initial transformation on each individual micrograph or solving the initial transformation with the whole tilt series will lead to the same result. To save the memory and keep the solution simple, in our implementation, the initial transformation for each micrograph in tilt series B is estimated individually.

S1.4 Sparse modeling of projection parameter estimation

An L-2 norm objective function is used to optimize the parameters:

$$E = \sum_i \sum_j \left(\mathrm{Proj}_i(\boldsymbol{X}_j) - \boldsymbol{x}_{i,j}\right)^2 \delta_{i,j}, \tag{27}$$

where $\mathcal{X} = \{\boldsymbol{X}_j\}$ is the set of spatial points to be estimated. $\mathrm{Proj}(\cdot)$ is the process of projection defined in Eq.4 and Eq.5. $\boldsymbol{x}_{i,j}$ is the configured position of fiducial marker and $\delta_{i,j}$ $(\delta_{i,j} \in \{0, 1\})$ indicates whether the j-th marker is visible in the i-th micrograph.

To practically solve the above estimation, let all parameters (projection parameters and spatial points) be converted into vector $\boldsymbol{p} \in \mathbb{R}^m$, and let f be a mapping that maps $\boldsymbol{p}$ to a vector $\hat{\boldsymbol{x}} \in \mathbb{R}^n$, $\hat{\boldsymbol{x}} = f(\boldsymbol{p})$. Thus, f denotes the projection operation, and $\hat{x}$ denotes all the predicted projection points (vector $\boldsymbol{x} \in \mathbb{R}^n$ denotes the measured projection points). The cost function E can be reformulated as $E = \sum_{i,j} \|\varepsilon_{i,j}\|^2$, where $\varepsilon = \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|$ ($\| \bullet \|$ denotes L2-norm) is the residual. Obviously, the original problem has been formulated in a nonlinear least-square style. Levenberg-Marquardt (L-M) algorithm can be utilized to solve the parameter optimization problem. The L-M algorithm approximates f in its neighborhood $\boldsymbol{p}$ as $f(\boldsymbol{p} + \boldsymbol{\sigma_p}) \approx f(\boldsymbol{p}) + \boldsymbol{J}\boldsymbol{\sigma_p}$ when $\|\boldsymbol{\sigma_p}\|$ is small. $\boldsymbol{J}$ is the Jacobian matrix of f. Note that the number of measured projection points is much larger than that of calibrated parameters. Thus, $m \ll n$, and the Jacobian matrix $\boldsymbol{J}$ is sparse.

In particular, we solve the following equations iteratively:

$$\boldsymbol{\Phi} = (\boldsymbol{J}^T\boldsymbol{J} + \mu\boldsymbol{I})^{-1}\boldsymbol{J}^T\boldsymbol{r} \tag{28}$$

where $\boldsymbol{\Phi} = (\boldsymbol{\Theta}, \boldsymbol{X})$, $\boldsymbol{\Theta}$ indicates projection parameters, $\boldsymbol{X}$ is the spatial points and $\boldsymbol{r}$ is the residual (i.e., difference between measured image points and estimated value over the projection process). $\boldsymbol{J} = \partial \boldsymbol{r} / \partial \boldsymbol{\Theta}$. $\boldsymbol{J}$ is the $M \times N$ Jacobian matrix, M is the number of measured image points and $N = n_\Theta + n_X$, which is the sum of the number of projection parameter numbers and spatial points.

Solving Eq.28 directly is possible but ineffective. However, the abundant zero values in $\boldsymbol{J}$ could be utilized, because some spatial points only have a few projection points. Consider the structure of $\boldsymbol{J}^T\boldsymbol{J}$,

$$\boldsymbol{J}^T\boldsymbol{J} = \begin{pmatrix} \frac{\partial \boldsymbol{r}^T}{\partial \boldsymbol{\Theta}} \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\Theta}} & \frac{\partial \boldsymbol{r}^T}{\partial \boldsymbol{\Theta}} \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{X}} \\ \frac{\partial \boldsymbol{r}^T}{\partial \boldsymbol{X}} \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{\Theta}} & \frac{\partial \boldsymbol{r}^T}{\partial \boldsymbol{X}} \frac{\partial \boldsymbol{r}}{\partial \boldsymbol{X}} \end{pmatrix} = \begin{pmatrix} \boldsymbol{C}_\Theta^{-1} & \boldsymbol{C}_{\Theta X}^{-1} \\ \boldsymbol{C}_{X\boldsymbol{\Theta}}^{-1} & \boldsymbol{C}_X^{-1} \end{pmatrix}, \tag{29}$$

where

$$\boldsymbol{C}_\Theta^{-1} = \begin{pmatrix} \sum_j \frac{\partial \boldsymbol{r}_{1j}}{\partial \boldsymbol{\Theta}_1}^T \frac{\partial \boldsymbol{r}_{1j}}{\partial \boldsymbol{\Theta}_1} & 0 & \cdots \\ 0 & \sum_j \frac{\partial \boldsymbol{r}_{2j}}{\partial \boldsymbol{\Theta}_2}^T \frac{\partial \boldsymbol{r}_{2j}}{\partial \boldsymbol{\Theta}_2} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}, \tag{30}$$

$$\boldsymbol{C}_X^{-1} = \begin{pmatrix} \sum_i \frac{\partial \boldsymbol{r}_{i1}}{\partial \boldsymbol{X}_1}^T \frac{\partial \boldsymbol{r}_{i1}}{\partial \boldsymbol{X}_1} & 0 & \cdots \\ 0 & \sum_i \frac{\partial \boldsymbol{r}_{i2}}{\partial \boldsymbol{X}_2}^T \frac{\partial \boldsymbol{r}_{i2}}{\partial \boldsymbol{X}_2} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}, \tag{31}$$

$$\boldsymbol{C}_{\Theta X}^{-1} = \begin{pmatrix} \frac{\partial \boldsymbol{r}_{11}}{\partial \boldsymbol{\Theta}_1}^T \frac{\partial \boldsymbol{r}_{11}}{\partial \boldsymbol{X}_1} & \frac{\partial \boldsymbol{r}_{12}}{\partial \boldsymbol{\Theta}_2}^T \frac{\partial \boldsymbol{r}_{11}}{\partial \boldsymbol{X}_1} & \cdots \\ \frac{\partial \boldsymbol{r}_{21}}{\partial \boldsymbol{\Theta}_1}^T \frac{\partial \boldsymbol{r}_{21}}{\partial \boldsymbol{X}_1} & \frac{\partial \boldsymbol{r}_{22}}{\partial \boldsymbol{\Theta}_2}^T \frac{\partial \boldsymbol{r}_{22}}{\partial \boldsymbol{X}_2} & \cdots \\ \vdots & \vdots & \ddots \end{pmatrix}. \tag{32}$$

We found that $\boldsymbol{C}_\Theta^{-1}$ and $\boldsymbol{C}_X^{-1}$ are both diagonal matrices. By substituting expressions from Eq.30 to Eq.32, Eq.28 can be rewritten as the following linear equation:

$$\begin{pmatrix} \boldsymbol{A} & \boldsymbol{B} \\ \boldsymbol{B}^T & \boldsymbol{C} \end{pmatrix} \begin{pmatrix} \boldsymbol{\Theta} \\ \boldsymbol{X} \end{pmatrix} = \begin{pmatrix} \boldsymbol{e}_\Theta \\ \boldsymbol{e}_X \end{pmatrix}, \tag{33}$$

where $\boldsymbol{A} = \boldsymbol{C}_\Theta^{-1} + \mu \boldsymbol{I}$, $\boldsymbol{C} = \boldsymbol{C}_X^{-1} + \mu \boldsymbol{I}$, $\boldsymbol{B} = \boldsymbol{C}_{\Theta X}^{-1}$, $\boldsymbol{e}_\Theta = \frac{\partial \boldsymbol{r}^T}{\partial \boldsymbol{\Theta}} r$ and $\boldsymbol{e}_X = \frac{\partial \boldsymbol{r}^T}{\partial \boldsymbol{X}} r$. A further term of Eq.33 is

$$\begin{pmatrix} \boldsymbol{A} - \boldsymbol{B}\boldsymbol{C}^{-1}\boldsymbol{B}^T & 0 \\ \boldsymbol{B}^T & \boldsymbol{C} \end{pmatrix} \begin{pmatrix} \boldsymbol{\Theta} \\ \boldsymbol{X} \end{pmatrix} = \begin{pmatrix} \boldsymbol{e}_\Theta - \boldsymbol{B}\boldsymbol{C}^{-1}\boldsymbol{e}_X \\ \boldsymbol{e}_X \end{pmatrix}. \tag{34}$$

Therefore, $\boldsymbol{\Theta}$ can be solved from the top half of Eq.34, and $\boldsymbol{X}$ can be solved from the bottom half. By employing the sparse linear problem, we reduce the scale of the problem.

S1.5 Additional choice for the projection model

Based on the non-linear projection theory proposed by (Lawrence *et al.*, 2006), a more complex projection model is possible to achieve better distortion correction. However, the complex projection model still has the limitation of model bias and risk of overfitting. AuTom-dualx provides the additional option “-w 2” for *dualxmauto* to achieve the following projection model estimation:

$$\begin{aligned} u &= a_0 + a_1X + a_2Y + a_3Z + a_4X \cdot Y + a_5X \cdot Z + a_6Y \cdot Z + a_7X \cdot X + a_8Y \cdot Y + a_9Z \cdot Z, \\ v &= b_0 + b_1X + b_2Y + b_3Z + b_4X \cdot Y + b_5X \cdot Z + b_6Y \cdot Z + b_7X \cdot X + b_8Y \cdot Y + b_9Z \cdot Z. \end{aligned} \tag{35}$$

In implementation, the estimation will be based on the result of extended projection model with w_x, w_y, w_z. However, as the application of the complex model has the risk of overfitting, the users should ensure that the number of fiducial markers is abundant enough (for statistical accuracy, we should have at least two or three fiducial markers for each coefficient to be determined), the quality of fiducial markers is high enough and the distribution of the fiducial markers is good enough.

S1.6 Technological difference between AuTom and IMOD

The main difference between the workflow of AuTom and IMOD is that AuTom uses a globally consistent alignment for the two tilt series but IMOD uses separate alignment, reconstruction and merging strategy. In addition, there are also other detailed differences between AuTom and IMOD.

Firstly, AuTom does not need the operation of coarse alignment based on cross-correlation but IMOD requires such an operation. Though the coarse alignment based on cross-correlation is very fast, it is not so robust against large image shift and rotation[2]. In IMOD, the fiducial marker detection is based on peak picking and hard threshold, while AuTom uses a sampling and clustering strategy. IMOD firstly detects the fiducial markers around the zero tilt series and then extends the searching to high-tilt micrographs by nearest neighbor search, based on the coarse alignment results. AuTom directly detects the fiducial markers of each micrograph with a refinement of each micrograph's average template. Therefore, AuTom is able to detect much more potential fiducial markers along the tilt series. In IMOD, the fiducial marker tracking is based on a pre-reconstruction of spatial fiducial markers from the low-tilt micrographs and the consequent nearest neighbor search, while AuTom utilizes the information of fiducial marker distribution to estimate a potential transformation of the fiducial marker positions from different micrographs. Here, IMOD's fiducial marker tracking highly depends on the initial detection of fiducial markers and coarse alignment results, whereas the performance of AuTom will not be affected by the initial positions of micrographs. Theoretically, AuTom will generate more fiducial marker tracks compared to IMOD but takes slightly longer time ($0.1 \sim 1$ seconds for each micrograph). However, IMOD's detection and tracking is not so stable and usually needs human intervention.

For alignment, AuTom uses a two-stage refinement of projection parameters while IMOD refines the projection parameters for each tilt series separately. In AuTom, sparse bundle adjustment is used to accelerate the optimization procedure. Therefore, for alignment, the first stage of AuTom is faster than IMOD ($\leqslant 20$ seconds). However, the second stage of AuTom is slower, which may take about $10 \sim 30$ minutes for a system that contains about 40 fiducial markers, because more projection parameters are used. For IMOD, a process called "solve match" and "combine" is used in the combination of the reconstruction from two different tilt series. The "solve match" will generate the transformation matrix for global and localized volumes to be used by the further combination. The "combine" will reweight the information of reconstruction and produce a summarized volume (about $10 \sim 50$ minutes for a system with 2048×2048 size). Here, the "solve match" in IMOD uses direct least square estimation of the spatial fiducial markers, while AuTom uses a two-step estimation (affine transformation from 3D point set registration result and rigid transformation approximation decomposed from affine transformation). Because the least square estimation is not robust to outliers, IMOD usually requires human intervention to generate an acceptable combination matrix. In AuTom, the transformation matrix and combination of the two tilt series are processed as described in Section S1.3. Here, AuTom does not need the volume combination stage. Due to the global projection parameters, AuTom can produce better alignment consistency between the two tilt series and result in a global reconstruction.

S2 Simulation details

S2.1 Tilt series generation

Because InSilicoTEM (Vulović *et al.*, 2013) does not consider fiducial markers, we used a modified InSilicoTEM scheme to generate the simulated dataset (Simul)[3]. The simulated tomogram with $2560 \times 2560 \times 250$ voxel that contains 81 volumes of 2WRJ was generated (2WRJ is the default MAP in InSilicoTEM, which is downsampled by 2 in our simulation). About 45 fiducial markers with a diameter of 18 pixels were embedded. From the tomogram, 51 projections were produced for each tilt axis (+50.0° to −50.0° at 2° intervals).

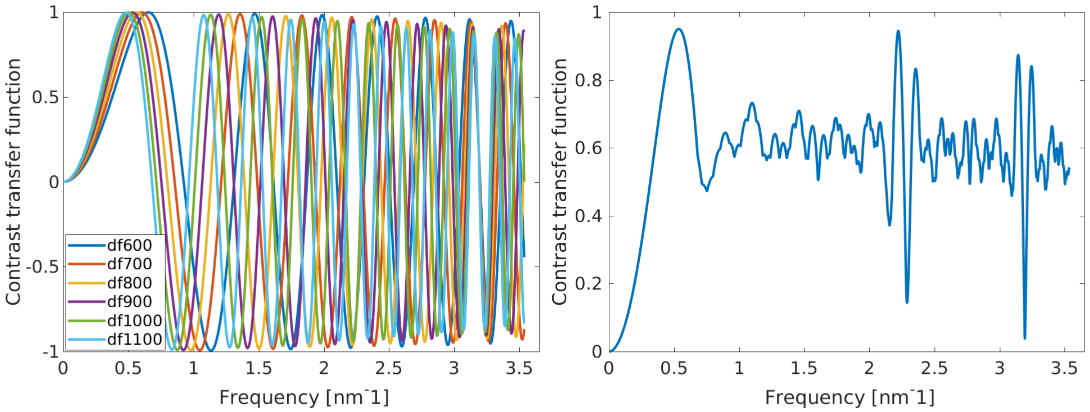
[2] https://drive.google.com/file/d/1uCnuCO2VwbAfFB97H7t5s94nRBxdtlZa/view?usp=sharing provides an example of how large image shift and rotation fails the coarse alignment in IMOD and the consequent process.

[3] The modified code can be download at https://drive.google.com/file/d/1i9nxRbY5lbfXS-gU2-yjBAj_IW3CuIuC/view?usp=sharing. Considering the radio damage caused by the electron beam, it is not natural to use dual-axis tomogram in subtomogram averaging. However, the conclusion drawn from the simulation is universal from the aspect of computational methodology.

One problem is that the CTF will affect the projected micrographs. It usually needs a set of tilt series with different defocus values and thousands of subtomograms to gain a discernible result. However, our aim is to demonstrate the quality improvement in the simultaneous reconstruction by the fashion of subtomogram averaging, instead of demonstrating the applicability of dual-axis reconstruction in subtomogram averaging. Here, to save time but without loss of generality, we tried to generate a set of projections with different defocus and then produced a noise-stained, phase-flipped averaging of the projections (similar to the 2D particle information averaging in SPA). Therefore, the simulated projections used here are phase-corrected and contain the Fourier information in different frequencies. Finally, random translation and rotation that range in ±150 pixels and ±10° were added to make the simulation more similar to an experimental dataset. The parameters used for simulation are listed below:

```
% ———————— General processing parameters (proc field)
params.proc.N = 2560; % Image size (field of view)
params.proc.partNum = 81; % Number of particles.
% ———————— Specimen (spec field)
params.spec.source = ’map’; % Options: ’map’, ’pdb’, or ’amorph’
params.spec.mapsample = ’2WRJ_VoxSize2.0A.mrc’;
params.spec.potcontribution = ’iasa’; % Potential type. Options: ’iasa’ or ’iasa+pb’
params.spec.motblur = 0; % Motion blur in [A]
params.spec.thick = 50e-09; % Thickness of the specimen[m]
params.spec.imagpot = 0; % Amplitude contrast flag. Options: (=0, none) (=1 constant Q)...
% ———————— Electron-specimen interaction (inter field)
params.inter.type = ’pa’; % Options: ’pa’ (projection), ’wpoa’ (weak-phase)...
params.inter.msdz = 1e-9; % Approximate thickness of the slice for multislice [m]...
% ———————— Microscope (mic field)
params.mic.Cs = 2.7e-3; % Spherical aberration [m]
params.mic.a_i = 0.030e-3; % Illumination aperture [rad]
params.mic.C_c = 2.7e-3; % Chromatic aberration [m]
params.mic.deltaE = 0.7; % Energy spread of the source [eV]
% ———————— aperture
params.mic.diam_obj = 100e-6; % Diameter of objective aperture [m]
params.mic.foc = 4.7e-3; % Focal distance [m]
% ———————— ideal phase plate (optional)
params.mic.PPflag = 0; % Phase plate flag
%———————— Acquisition settings (acquis field)
params.acquis.pixsize = 0.200e-9; % Pixel size in the specimen plane [m]3748
% The interpretation of defocus has been changed in our modified codes
params.acquis.df = [600,700,800,900,1000,1100]*1e-9; % Note: undefocus df> 0; overfocus df< 0
params.acquis.ast = [0]*1e-9; % Astigmatism [m].
params.acquis.astangle = 0*pi/180; % Astigmatism angle [rad]
params.acquis.Voltage = 300e3; % Acceleration voltage
params.acquis.tilt = [-50:2:50]/180*pi; % Tilt geometry
params.acquis.dose_on_sample = [50]/length(params.acquis.tilt); % Integrated flux [e-/A2]
% ———————— Detector-Camera (cam field)
params.cam.type = ’ideal’; % Options: ’Eagle4k’, ’US4000’,...’ideal’ (100% at Nq)
params.cam.bin = 1; % hardware binning
params.cam.GenMTFasEMG = 1;
params.cam.DQEflag = 0; % Flag for dqe (=0 means NTF = MTF)
```

Considering the modulation of CTF, we generated a set of projections with defocus range from 600e-9 m to 1100e-9 m and then averaged the phase-flipped projections with different defocus. Figure S4 demonstrates the effects of CTF. Our final experiment is based on the obtained average projections, in which the Poisson and Gaussian noise were added according to the configuration of the camera.



(A)

(B)

Figure S4: Illustration of the CTF curves with defocus from 600e-9 to 1100e-9 (m). (A) The original CTF curves. (B) The phase flipped averaging.

S2.2 Reconstructed spatial fiducial marker comparison

The “glbpoint.mot” in AuTom-dualx records the final estimation of spatial fiducial markers, and the “_afid.xyz” in IMOD records the final estimation of spatial fiducial markers for tilt series A (or “_bfid.xyz” for tilt series B). For Simulation, the simulated spatial fiducial marker locations are known as the ground-truth. Due to the shift and rotation in tilt series alignment, the estimated spatial fiducial markers will have rigid transformation compared to the ground-truth. Here, for the fiducial markers recorded in “glbpoint.mot” and “xxxafid.xyz”, we firstly registered the estimated spatial fiducial markers to the ground-truth and then calculated the peer-to-peer deviations. In the demonstrated Simul dataset, AuTom-dualx reconstructed 45 spatial fiducial markers and IMOD reconstructed 44 spatial fiducial markers. Here, we compared the “glbpoint.mot” with the “Simulafid.xyz”. Figure S5 shows the histogram of the deviations between the estimated spatial fiducial markers and the ground-truth fiducial marker locations. Here, the average reprojection residual of IMOD is 0.292 for tilt series A and 0.299 for tilt series B, whereas that of AuTom-dualx is 0.201. And the average spatial deviation between IMOD’s results and the ground-truth is 0.1318, while the average spatial deviation between AuTom-dualx’s results and the ground-truth is 0.0511. These results demonstrate that AuTom-dualx produces better alignment than IMOD. The improvement may be due to two reasons: location improvement from fiducial marker detection (Han *et al.*, 2015) and the introduction of the dual-axis bundle adjustment model.

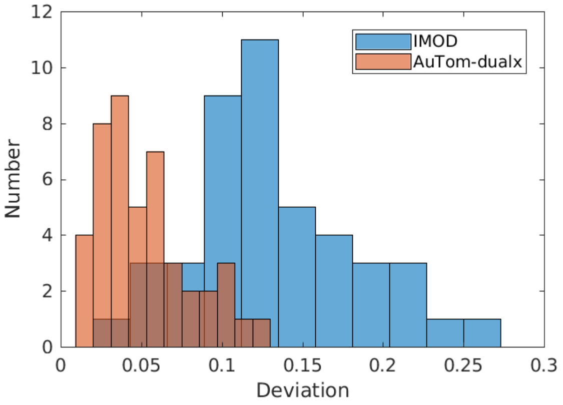


Figure S5: The histogram of the deviations between the estimated spatial fiducial markers and the ground-truth fiducial marker locations.

References

Aiger, D., Mitra, N. J., and Cohen-Or, D. (2008). 4-points congruent sets for robust pairwise surface registration. *ACM T. Graphic.*, **27**(3), 85:1–85:10.

Ankerst, M., Breunig, M. M., peter Kriegel, H., and Sander, J. (1999). Optics: Ordering points to identify the clustering structure. pages 49–60. ACM Press.

Bentley, J. L. (1975). Multidimensional binary search trees used for associative searching. *Commun. ACM*, **18**(9), 509–517.

Betzig, E., Patterson, G. H., Sougrat, R., Lindwasser, O. W., Olenych, S., Bonifacino, J. S., Davidson, M. W., Lippincott-Schwartz, J., and Hess, H. F. (2006). Imaging intracellular fluorescent proteins at nanometer resolution. *Science*, **313**(5793), 1642–1645.

Fischler, M. A. and Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, **24**(6), 381–395.

Han, R., Wang, L., Liu, Z., Sun, F., and Zhang, F. (2015). A novel fully automatic scheme for fiducial marker-based alignment in electron tomography. *J. Struct. Biol.*, **192**(3), 403 – 417.

Han, R., Zhang, F., and Gao, X. (2018). A fast fiducial marker tracking model for fully automatic alignment in electron tomography. *Bioinformatics*, **34**(5), 853–863.

Jian, B. and Vemuri, B. C. (2011). Robust point set registration using gaussian mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **33**(8), 1633–1645.

Lawrence, A., Bouwer, J. C., Perkins, G., and Ellisman, M. H. (2006). Transform-based backprojection for volume reconstruction of large format electron microscope tilt series. *J. Struct. Biol.*, **154**(2), 144 – 167.

Marquardt, D. W. (1963). An algorithm for least-squares estimation of nonlinear parameters. *Journal of the society for Industrial and Applied Mathematics*, **11**(2), 431–441.

Myronenko, A. and Song, X. (2010). Point set registration: Coherent point drift. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **32**(12), 2262–2275.

Nieuwenhuizen, R. P., Lidke, K. A., Bates, M., Puig, D. L., Grünwald, D., Stallinga, S., and Rieger, B. (2013). Measuring image resolution in optical nanoscopy. *Nature methods*, **10**(6), 557–562.

Vulović, M., Ravelli, R. B., van Vliet, L. J., Koster, A. J., Lazić, I., Lücken, U., Rullgård, H., Öktem, O., and Rieger, B. (2013). Image formation modeling in cryo-electron microscopy. *Journal of structural biology*, **183**(1), 19–32.