Angle-Independent 3D Reconstruction

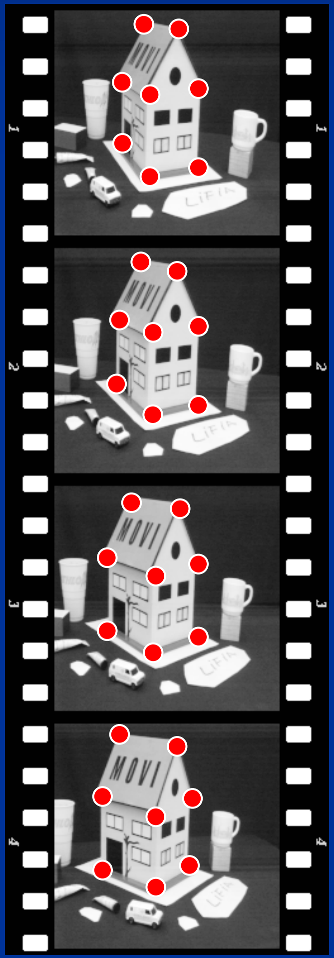
Ji Zhang

Mireille Boutin

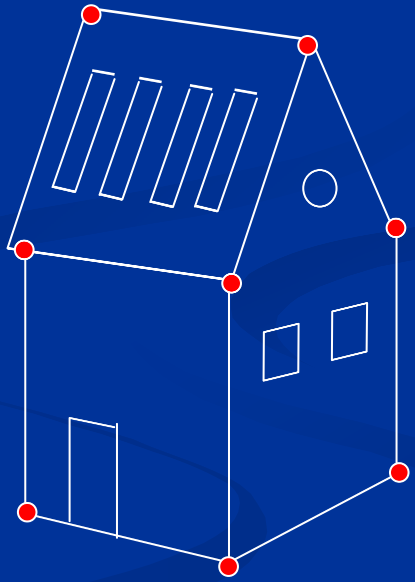
Daniel Aliaga

Goal: Structure from Motion

To reconstruct the 3D geometry of a scene from a set of pictures (e.g. a movie) of the scene



point reconstruction

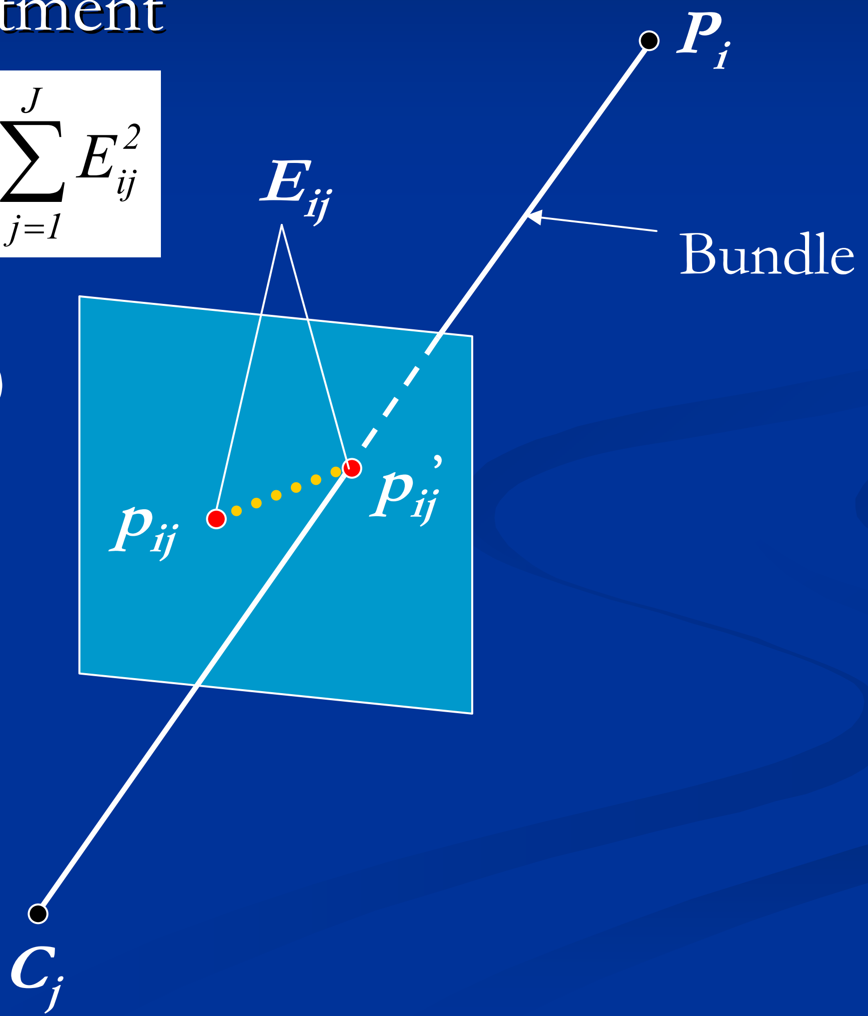


Previous Work

- Bundle Adjustment

Minimize: $$E = \sum_{i=1}^{N} \sum_{j=1}^{J} E_{ij}^2$$

Complexity: O(*JN*)



Previous Work

- Bundle Adjustment
  - Is a global optimization tolerant to missing samples and outliers.
  - Optimizes both structure and motion even though structure may be the only objective.
  - Needs a sufficiently good initial guess.
  - Usually used as a final refinement stage.

Previous Work

- Projective Reconstruction

  + upgrade to Euclidean Reconstruction

  - Use epipolar geometry to solve for a projective reconstruction ***P***, which equals to Euclidean reconstruction ***E*** multiplied by a linear transform ***F***.
  - Use extra information to get the linear transform ***F*** and upgrade the projective reconstruction ***P*** to Euclidean reconstruction ***E***.

$$P = E \cdot F$$

Previous Work

- Projective Reconstruction
  + upgrade to Euclidean Reconstruction
  - The first step is linear while the second one is nonlinear.
  - Structure reconstruction depends on motion reconstruction.
  - Sensitive to noise, need refinement (BA).

Traditional Projective Equations

$$\begin{pmatrix} p_{ij} \\ 1 \end{pmatrix} = c_{ij} F_j P_i,$$

for $i = 1, \ldots, N$ and $j = 1, \ldots, J$.

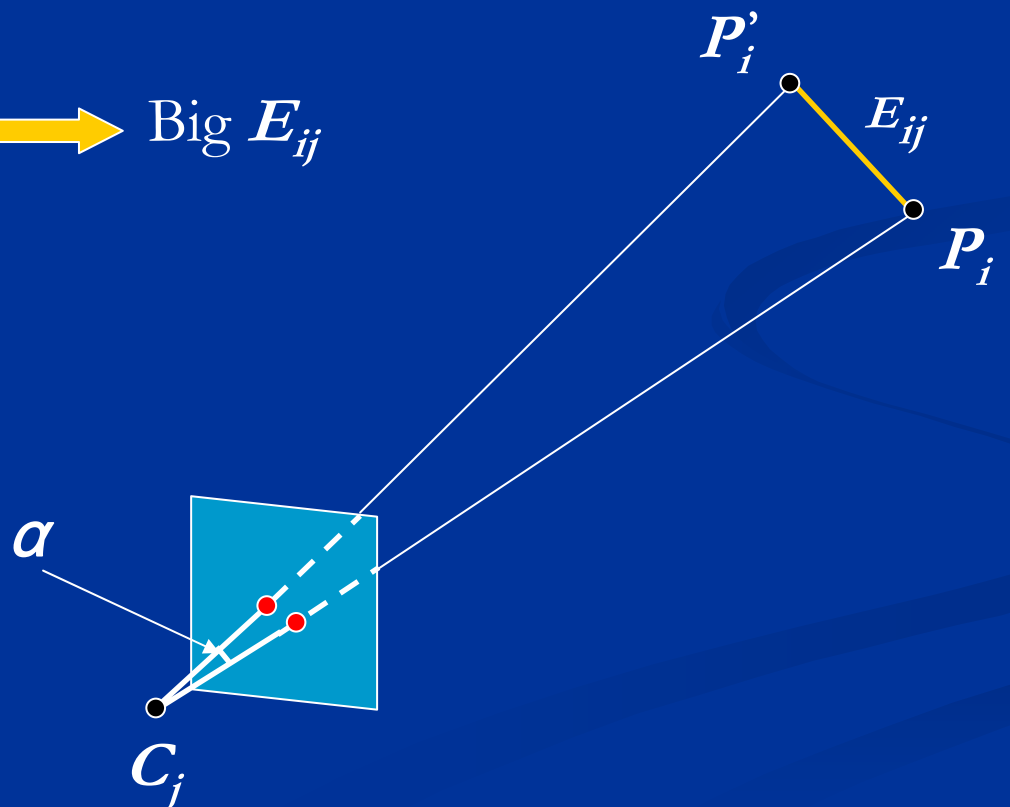
$$F_j = (R_j, T_j) \qquad p_{ij} = (x_{ij}, y_{ij})$$

where $\boldsymbol{p_{ij}}$ represents the 2D coordinates of the 3D feature point $\boldsymbol{P_i}$ observed on picture *j*, $\boldsymbol{c_{ij}}$ is a constant, and $\boldsymbol{F_j}$ is a 3-by-4 matrix containing the camera parameters corresponding to picture *j*.

Challenge

- Overcoming the fact that camera pose reconstruction is ill conditioned, especially for camera orientation

Small α → Big E_{ij}



Our Approach

- Variable elimination
  - Eliminate camera position and/or orientation from the reconstruction process
  - This is *not* self-calibration which computes the camera pose on its own; rather camera position and/or orientation is mathematically eliminated from the equations

Our Approach: Eliminate Camera Pose

Gaussian Elimination (easy)

$$c_{11}x_1 + c_{12}x_2 + c_{13}x_3 + c_{14} = 0$$
$$c_{21}x_1 + c_{22}x_2 + c_{23}x_3 + c_{24} = 0$$
$$c_{31}x_1 + c_{32}x_2 + c_{33}x_3 + c_{34} = 0$$

$$\downarrow$$

$$k_{11}x_1 + k_{12}x_2 + k_{13}x_3 + k_{14} = 0$$
$$k_{22}x_2 + k_{23}x_3 + k_{24} = 0$$
$$k_{33}x_3 + k_{34} = 0$$

Nonlinear Elimination (hard)

$$F_1(x_1, x_2, x_3) = 0$$
$$F_2(x_1, x_2, x_3) = 0$$
$$F_3(x_1, x_2, x_3) = 0$$

$$\downarrow$$

$$G_1(x_1, x_2, x_3) = 0$$
$$G_2(x_2, x_3) = 0$$
$$G_3(x_3) = 0$$

Our Approach: Eliminate Camera Pose

- Gröbner Bases

  Degree increases rapidly when eliminating variables.

- Invariant Based Elimination
  - Pierre-Louis Bazin, Mireille Boutin (2003)
  - Invariant theory
  - Moving frame

Invariant Based Elimination

$$\gamma_{ij}\gamma_{1j}k_{1ij} = (P_i - C_j)\cdot(P_1 - C_j),\ \text{for } i = 3,\ldots,N$$

$$\gamma_{ij}\gamma_{2j}\gamma_{1j}^2 k_{2ij} = ((P_1 - C_j)\times(P_i - C_j))\cdot((P_1 - C_j)\times(P_2 - C_j)),\ \text{for } i = 2,\ldots,N$$

$$\gamma_{ij}\gamma_{2j}\gamma_{1j}k_{3ij} = (P_i - C_j)\cdot((P_1 - C_j)\times(P_2 - C_j)),\ \text{for } i = 1,\ldots,N$$

$$k_{1ij} = ((x_{ij}, y_{ij}, -1)\cdot(x_{1j}, y_{1j}, -1)$$

$$k_{2ij} = ((x_{1j}, y_{1j}, -1)\times(x_{ij}, y_{ij}, -1))\cdot((x_{1j}, y_{1j}, -1)\times(x_{2j}, y_{2j}, -1))$$

$$k_{3ij} = (x_{ij}, y_{ij}, -1)\cdot((x_{1j}, y_{1j}, -1)\times(x_{2j}, y_{2j}, -1))$$

$$\gamma_{ij} = \frac{w_{ij}}{w_{ij} - w_{0j}} = \frac{\left\| P_i - C_j \right\|}{\left\| (x_{ij}, y_{ij}, -1) - (0,0,0) \right\|}$$

Invariant Based Elimination

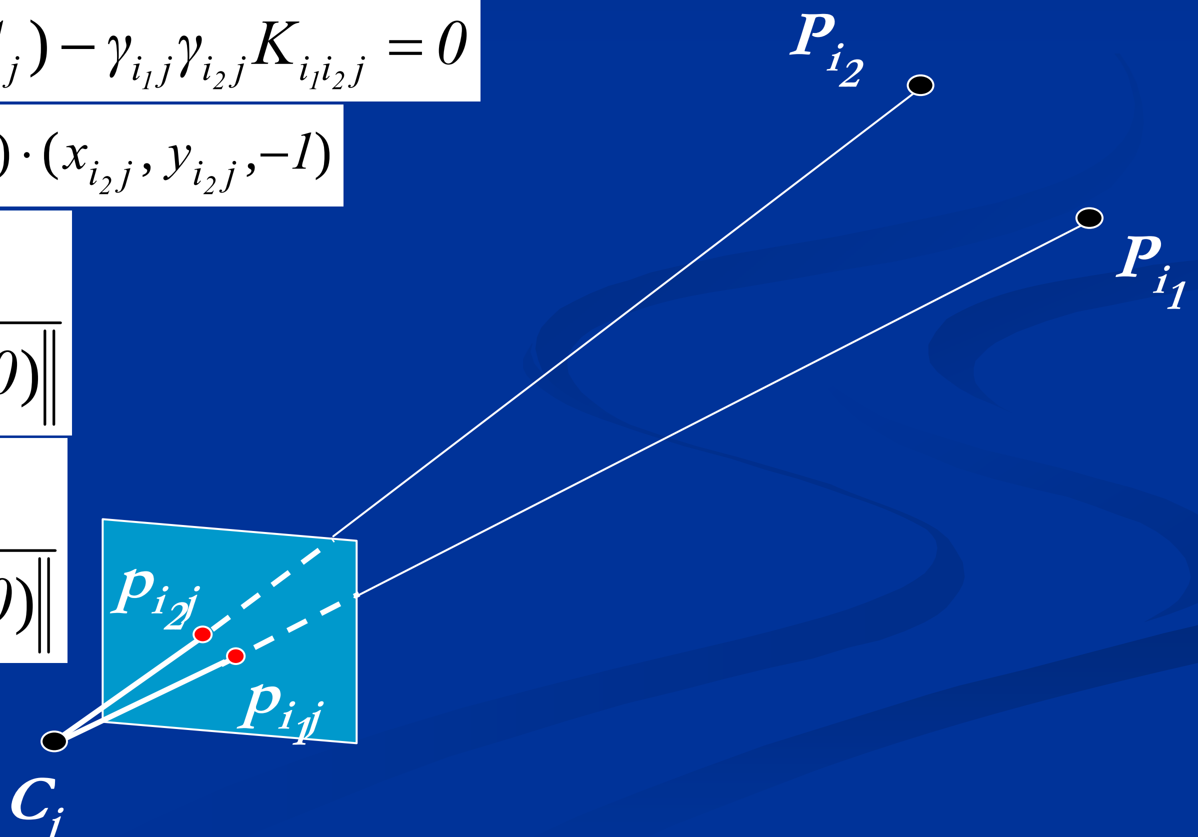
Simplify the equations to make it work for least square refinement. We focus on lower degree and symmetry.

$$(P_{i_1} - C_j)\cdot(P_{i_2} - C_j) - \gamma_{i_1 j}\gamma_{i_2 j}K_{i_1 i_2 j} = 0$$

$$K_{i_1 i_2 j} = (x_{i_1 j}, y_{i_1 j}, -1)\cdot(x_{i_2 j}, y_{i_2 j}, -1)$$

$$\gamma_{i_2 j} = \frac{\left\| P_{i_2} - C_j \right\|}{\left\| p_{i_2 j} - (0,0,0) \right\|}$$

$$\gamma_{i_1 j} = \frac{\left\| P_{i_1} - C_j \right\|}{\left\| p_{i_1 j} - (0,0,0) \right\|}$$



Invariant Based Elimination

A new angle independent cost function for Bundle Adjustment Refinement (AIBAR).

$$E = \sum_{i_1=1}^{N}\sum_{i_2=1}^{N}\sum_{j=1}^{J}\left[\left(P_{i_1} - C_j\right)\cdot\left(P_{i_2} - C_j\right) - \gamma_{i_1 j}\gamma_{i_2 j}K_{i_1 i_2 j}\right]^2$$

Here ***N*** is the number of 3D points, ***J*** is the number of images. Complexity $O(JN^2)$ higher than traditional BA $O(JN)$

Invariant Based Elimination

Observe the original equations, if we know $\boldsymbol{P_1}$, $\boldsymbol{P_2}$ and $\boldsymbol{C_j}$, then the system is linear.

$$
\begin{pmatrix}
M_1 & 0 & & 0 & V_{31} & 0 & & 0 \\
0 & M_1 & & 0 & 0 & V_{41} & & 0 \\
 & & \ddots & & & & \ddots & \\
0 & 0 & & M_1 & 0 & 0 & & V_{N1} \\
M_2 & 0 & & 0 & V_{32} & 0 & & 0 \\
0 & M_2 & & 0 & 0 & V_{42} & & 0 \\
 & & \ddots & & & & \ddots & \\
0 & 0 & & M_2 & 0 & 0 & & V_{N2} \\
 & \vdots & & & & \vdots & & \\
M_J & 0 & & 0 & V_{3J} & 0 & & 0 \\
0 & M_J & & 0 & & V_{4J} & & 0 \\
 & & \ddots & & & & \ddots & \\
0 & 0 & & M_J & 0 & 0 & & V_{NJ}
\end{pmatrix}
\begin{pmatrix}
P_3 \\ P_4 \\ \vdots \\ P_N \\ \gamma_{31} \\ \gamma_{41} \\ \vdots \\ \gamma_{NJ}
\end{pmatrix}
=
\begin{pmatrix}
B_1 \\ B_1 \\ \vdots \\ B_1 \\ B_2 \\ B_2 \\ \vdots \\ B_J
\end{pmatrix}
$$

$$
M_j = \begin{pmatrix}
P_1 - C_j \\
\|P_1 - C_j\|^2 (P_2 - C_j) - (P_1 - C_j)\cdot(P_2 - C_j)(P_1 - C_j) \\
(P_1 - C_j)\times(P_2 - C_j)
\end{pmatrix}
$$

$$
V_{ij} = \begin{pmatrix}
-\gamma_{1j} k_{1ij} \\
-\gamma_{2j}\gamma_{1j}^2 k_{2ij} \\
-\gamma_{2j}\gamma_{1j} k_{3ij}
\end{pmatrix}
$$

$$
B_j = \begin{pmatrix}
C_j \cdot (P_1 - C_j) \\
C_j \cdot (\|P_1 - C_j\|^2 (P_2 - C_j) - ((P_1 - C_j)\cdot(P_2 - C_j))(P_1 - C_j)) \\
C_j \cdot ((P_1 - C_j)\times(P_2 - C_j))
\end{pmatrix}
$$

Two Possible Applications

- Speed Priority:

  Select a number of pairs to be use as "anchor points".

  1. Reconstructed anchor points via initial guess and least squares minimization.

  2. Reconstructed then rest of points linearly.

- Accuracy Priority:

  Use least square minimization to refine all 3D points and camera centers (AIBAR).

Two Possible Applications

Objective:

To compare the robustness of our method with SBA (Sparse Bundle Adjustment).

Application 1

- Estimate camera center and camera orientation as best as we can.
- Add noise in camera orientation incrementally.

  a) Reconstruct structure using linear triangulation to generate initial guess.

  b) Use the result of a) as initial guess for SBA and our method .

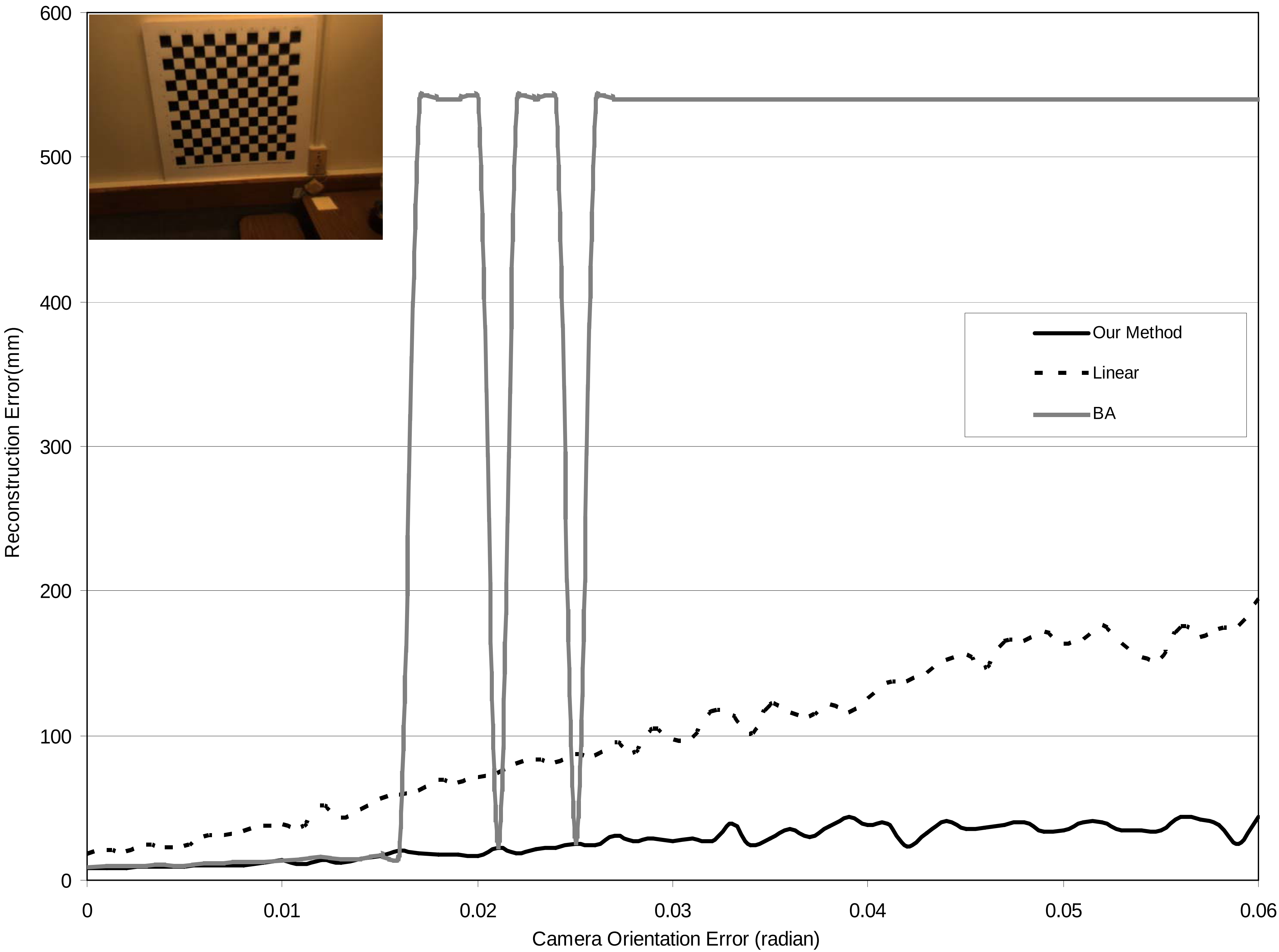
  c) Compare the results.

Experiments

1. Check Board

   number of points=96

   number of images=48

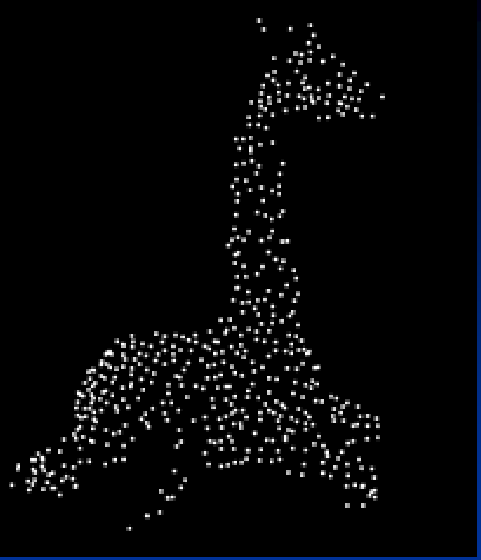

Reconstruction Error(mm)
600
500
400
300
200
100
0
0
0.01
0.02
0.03
0.04
0.05
0.06
Camera Orientation Error (radian)
Our Method
Linear
BA

Experiments

2. Giraffe

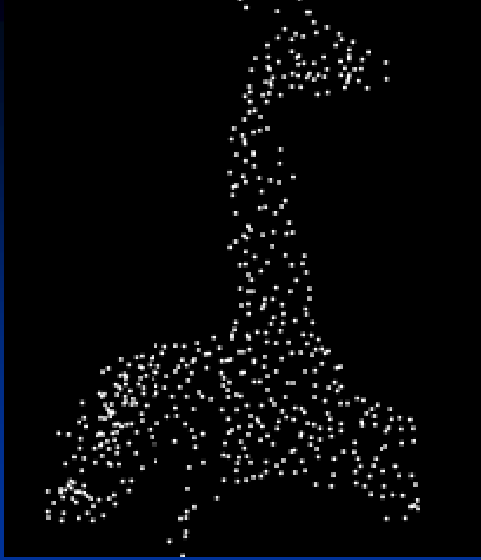
   number of points=480

   number of images=360



No noise

6 degrees noise



12 degrees noise

Actual model

Linear with 6 degrees noise

BA with 12 degrees noise

Experiments

3. House

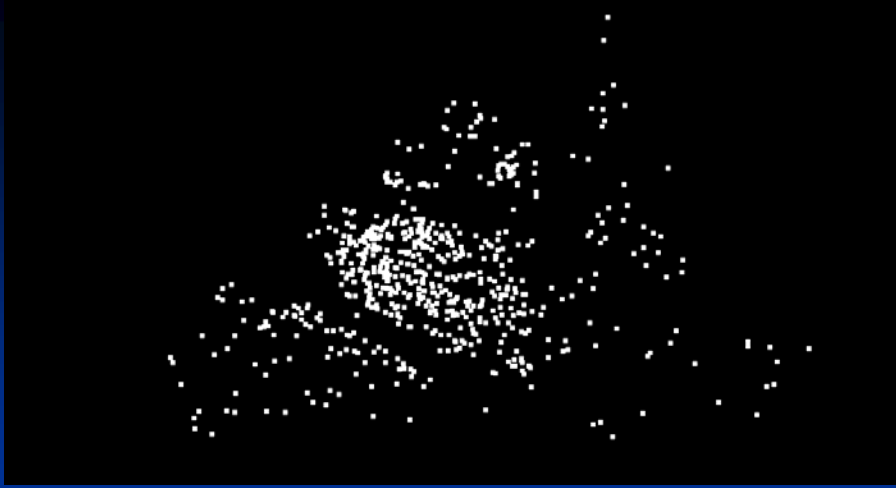
   number of points=672

   number of images=10

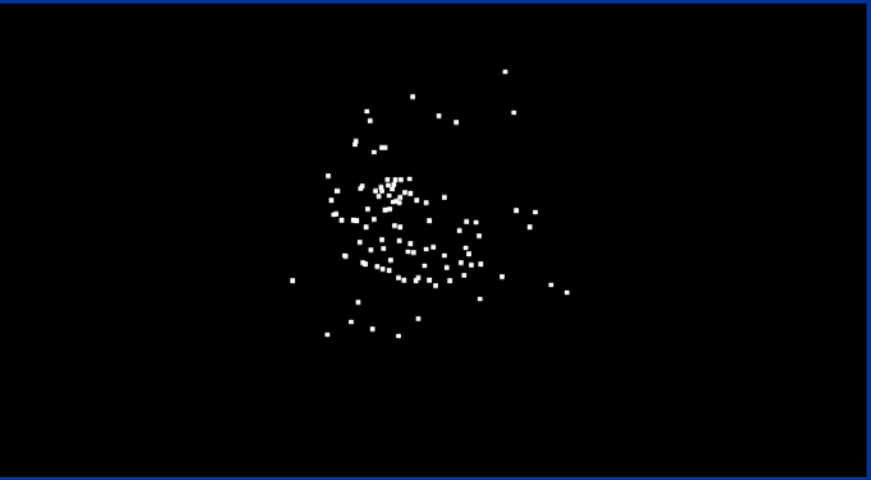
   Camera position error: 4% of the model space diagonal

   Camera orientation error: 7 degrees.

Actual model



Our method



Linear

BA

Application 2

- Estimate camera center, camera orientation, structure as best as we can.
- Add error in camera center, camera orientation, and structure incrementally.

  a) Refine structure+camera center+camera orientation using SBA.

  b) Refine structure+camera center using AIBAR.

Experiments

1. Check Board

   number of points=96

   number of images=48

   Time cost:

   AIBAR: 838s.

   SBA: 12s.

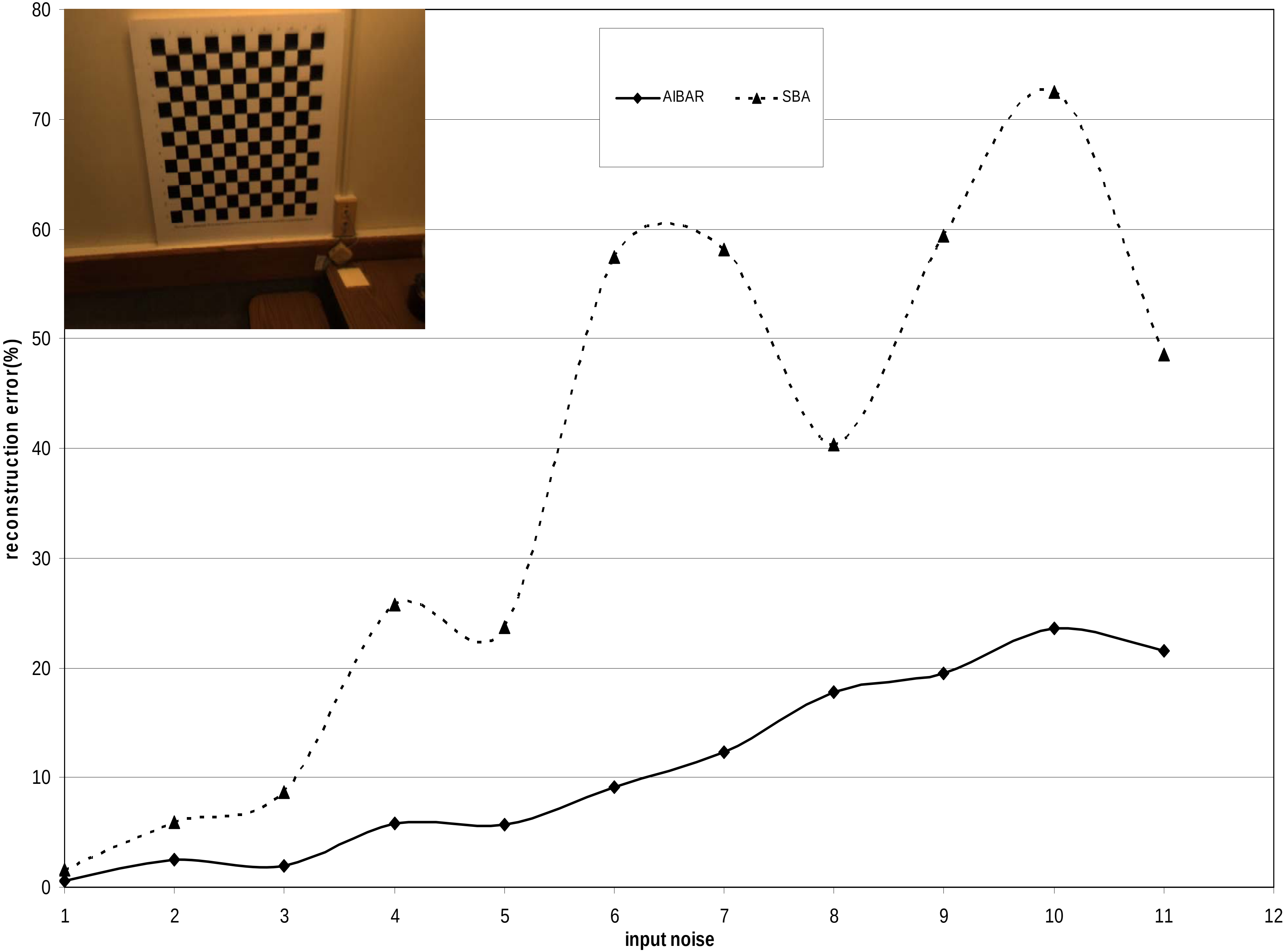

AIBAR
SBA
reconstruction error(%)
input noise
0
10
20
30
40
50
60
70
80
1
2
3
4
5
6
7
8
9
10
11
12

Image Sequence Partitioning

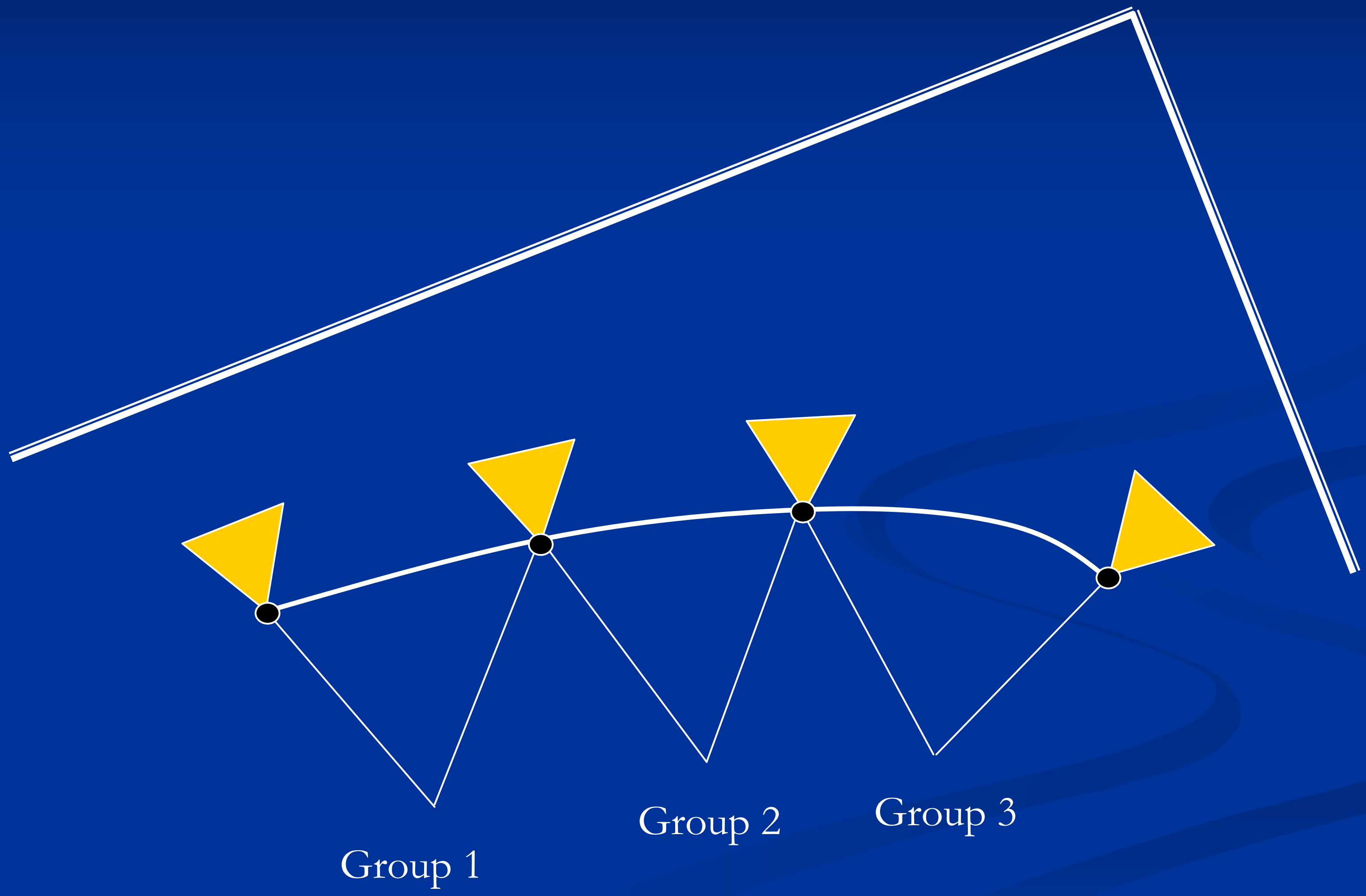
Complexity comparison:

Our: $O(JN^2)$ BA: $O(JN)$

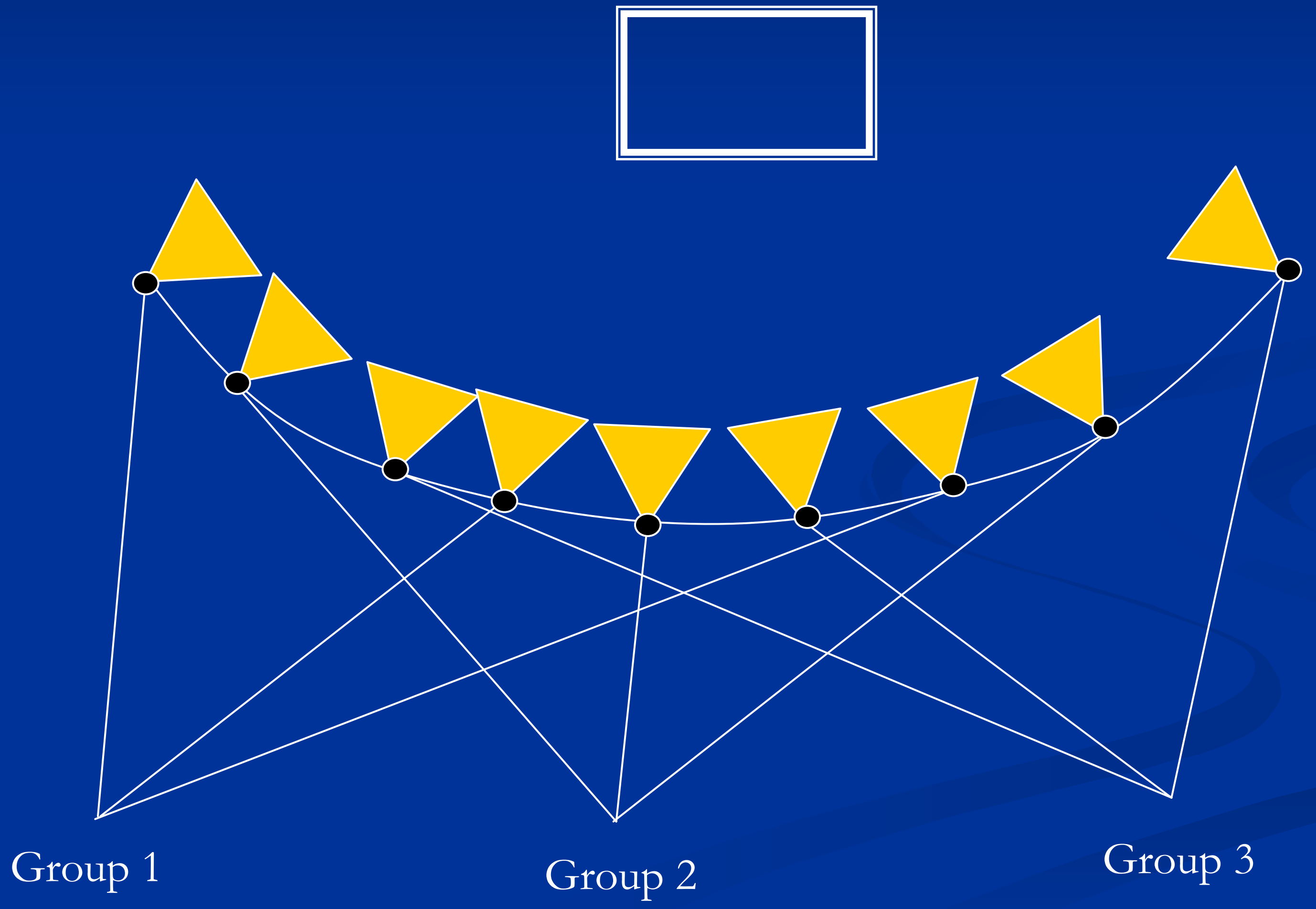
To compensate for the additional computational cost, subdivide the image sequence into disjoint subsets according to different type of sequence.

- Inside looking out sequence
- Outside looking in sequence

Inside looking out sequence



Outside looking in sequence



Experiments

1. Check Board

   number of points=96

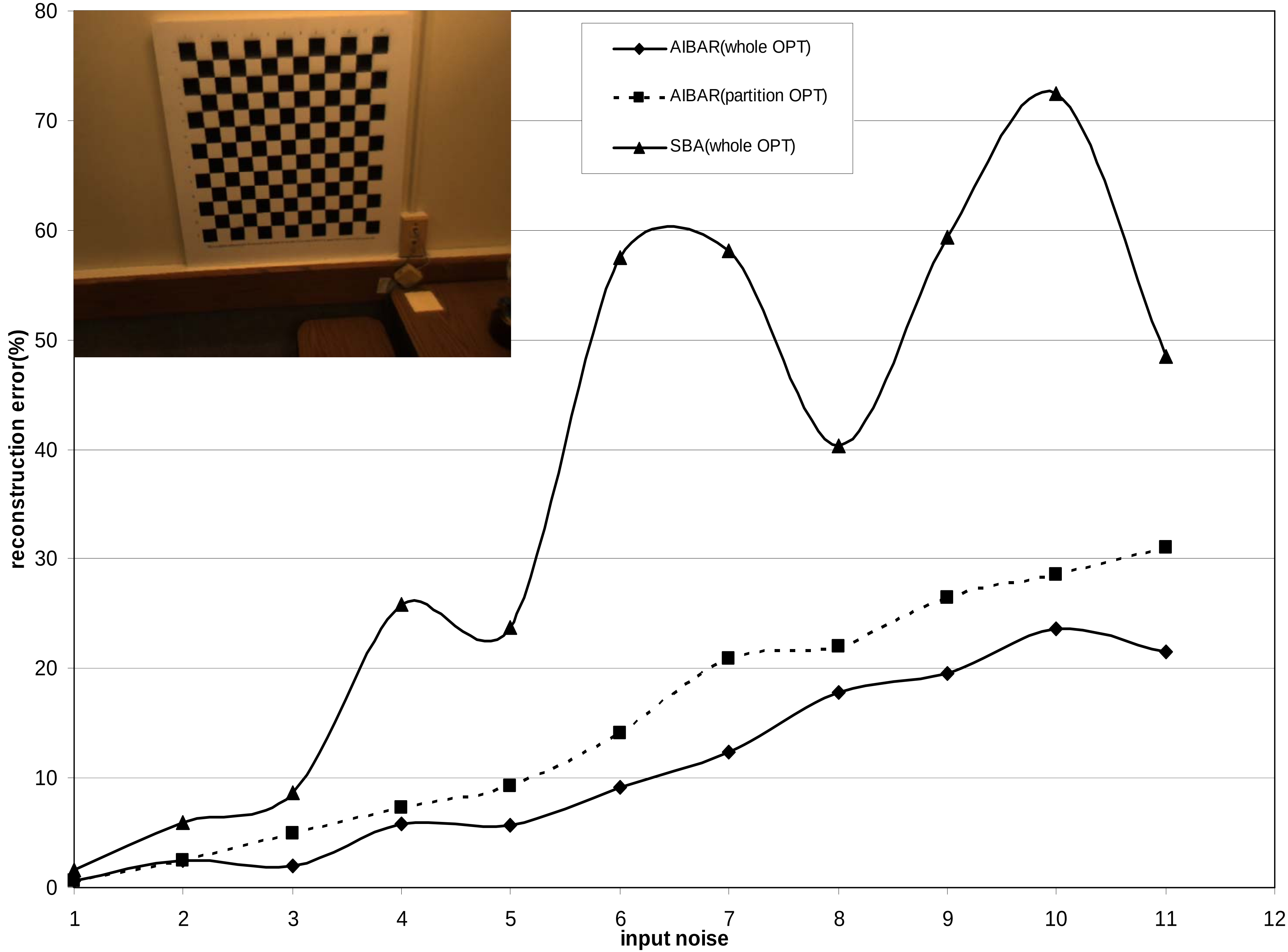
   number of images=48

   Time cost:

      AIBAR (without partition): 838s.

      AIBAR (with partition into 8 subsets): 14s.

      SBA: 12s.


AIBAR(whole OPT)
AIBAR(partition OPT)
SBA(whole OPT)
reconstruction error(%)
input noise
0
10
20
30
40
50
60
70
80
1
2
3
4
5
6
7
8
9
10
11
12

Experiments

2. Giraffe

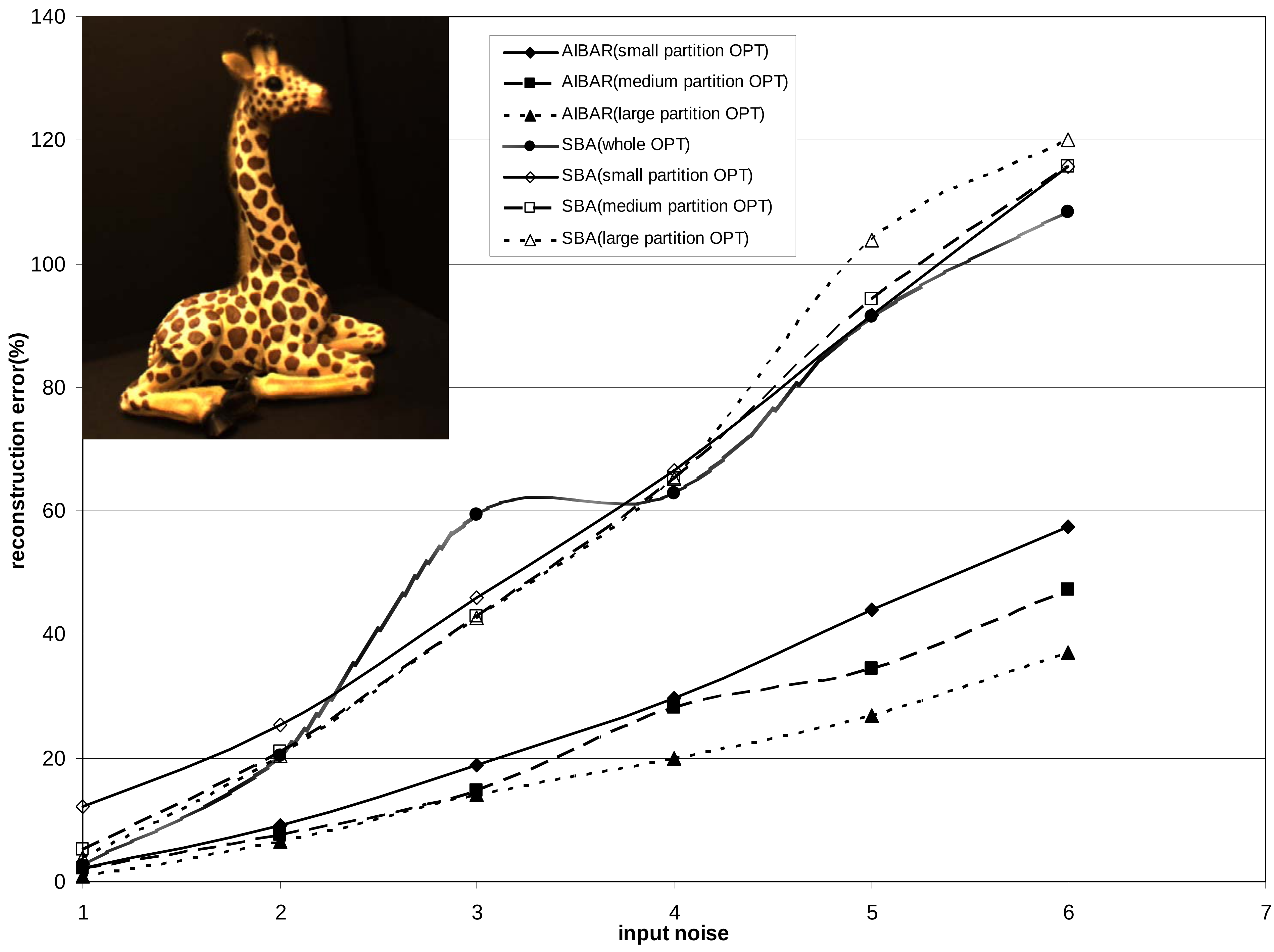
   number of points=480

   number of images=360

   Time cost for different partitions:

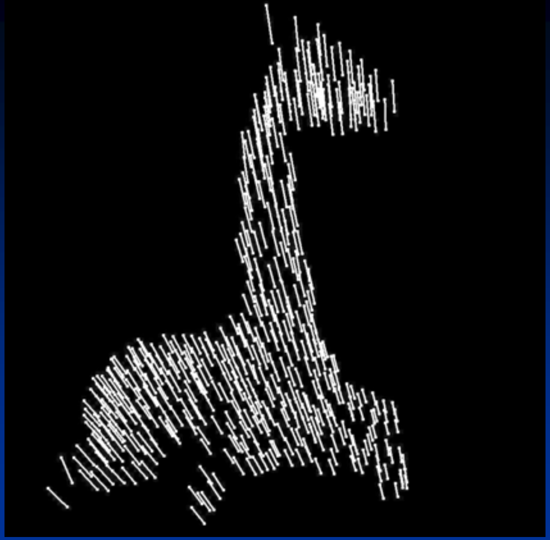
AIBAR	None	Small	Medium	Large
(s)		88	253	961

SBA	None	Small	Medium	Large
(s)	6198	7	12	43

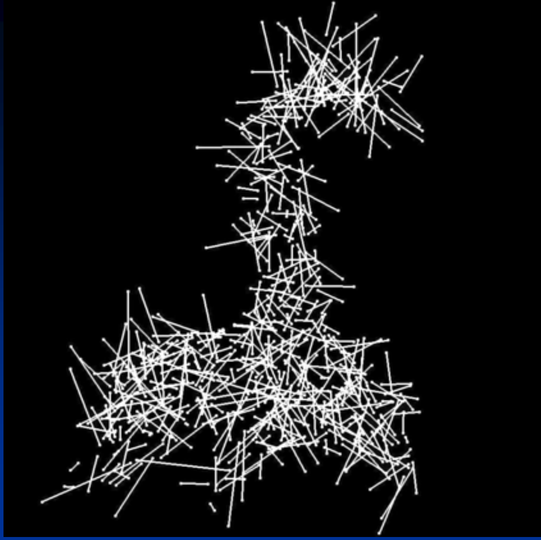


AIBAR(small partition OPT)
AIBAR(medium partition OPT)
AIBAR(large partition OPT)
SBA(whole OPT)
SBA(small partition OPT)
SBA(medium partition OPT)
SBA(large partition OPT)
reconstruction error(%)
input noise
0
20
40
60
80
100
120
140
1
2
3
4
5
6
7

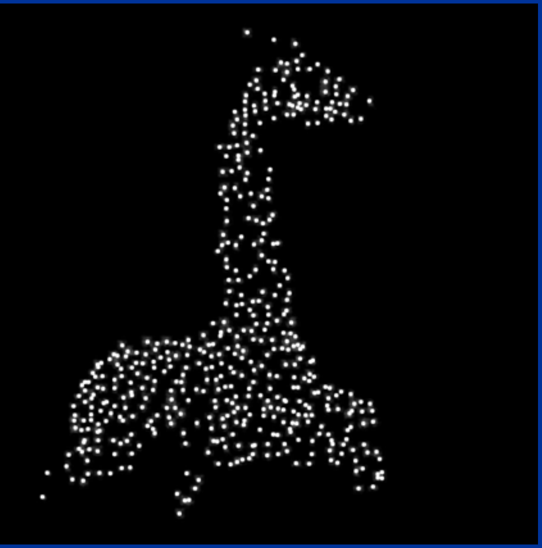
Actual Model



SBA (medium noise)

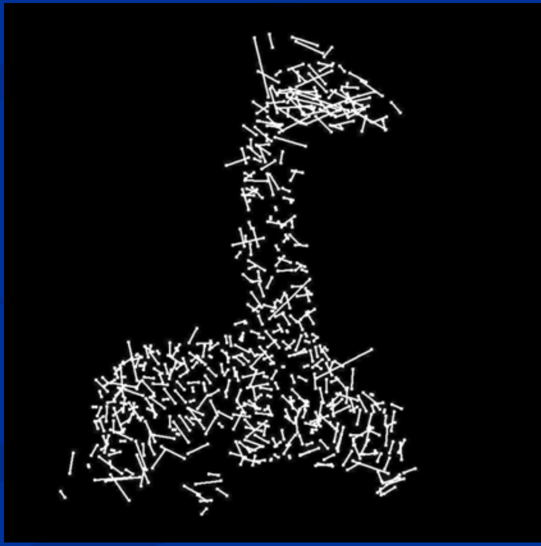


SBA (large noise)



AIBAR (no noise)

AIBAR (medium noise)



AIBAR (large noise)

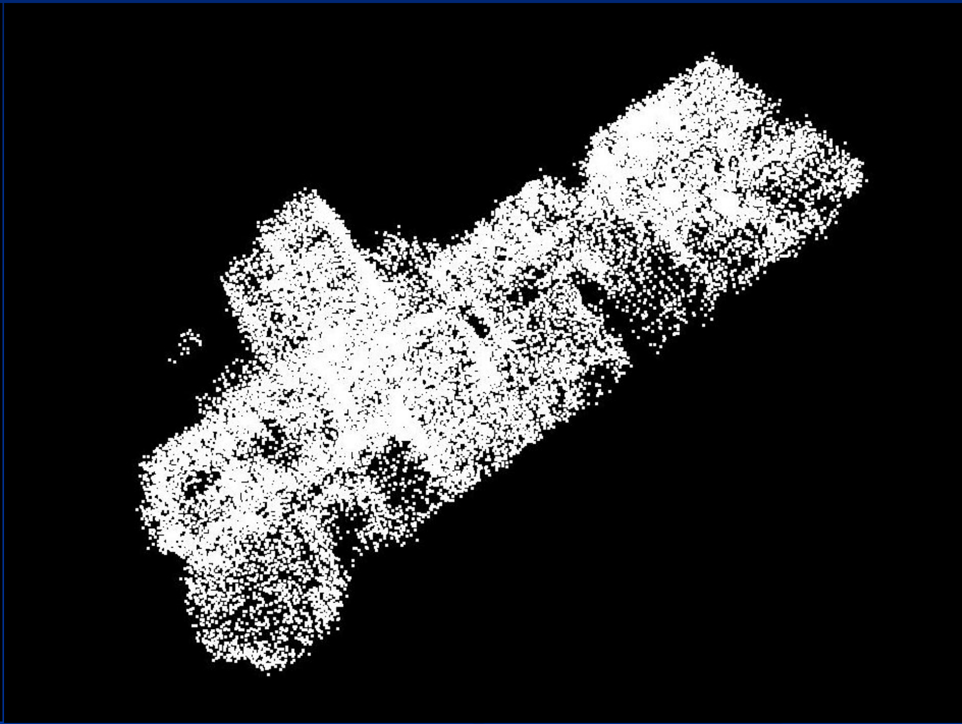
Experiments

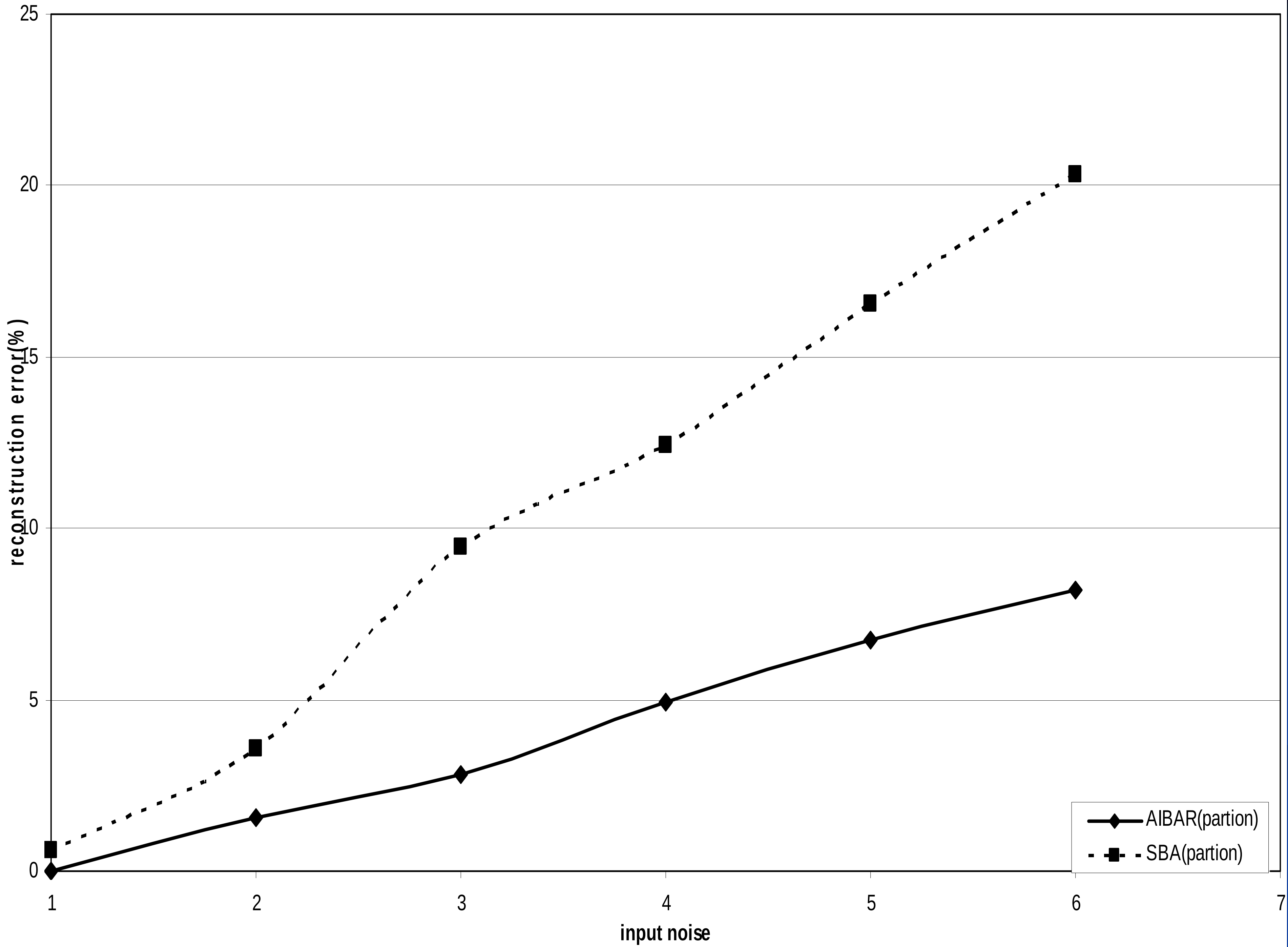
3. Floor

   number of points=32688

   number of images=2644

   Partition: 155 subsets





reconstruction error(%)
25
20
15
10
5
0
1
2
3
4
5
6
7
input noise
AIBAR(partion)
SBA(partion)

Conclusion

- Low dimension: 2.
- Robust : angle independent.
- Complexity: $O(JN^2)$.
- Scalable: partition or using anchor points.

Future Work

- Eliminate camera centers
- Lower the complexity.
- Improve the partitioning.

Thank you!