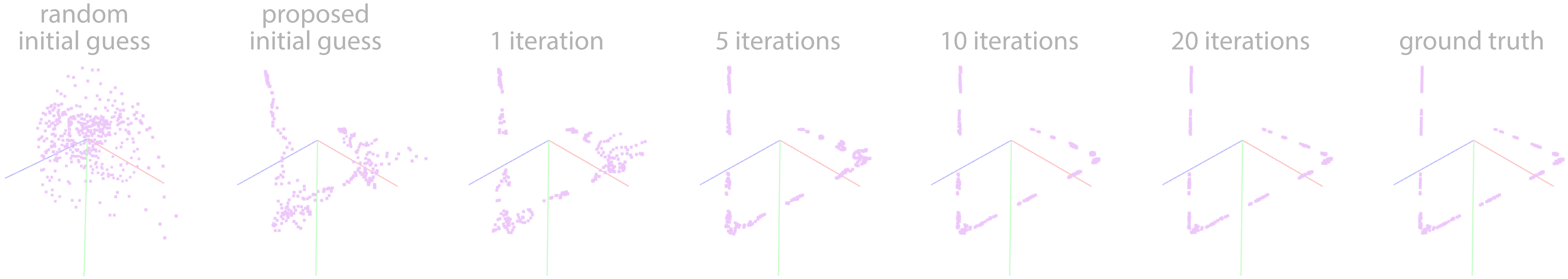


Hyperspectral Inverse Skinning

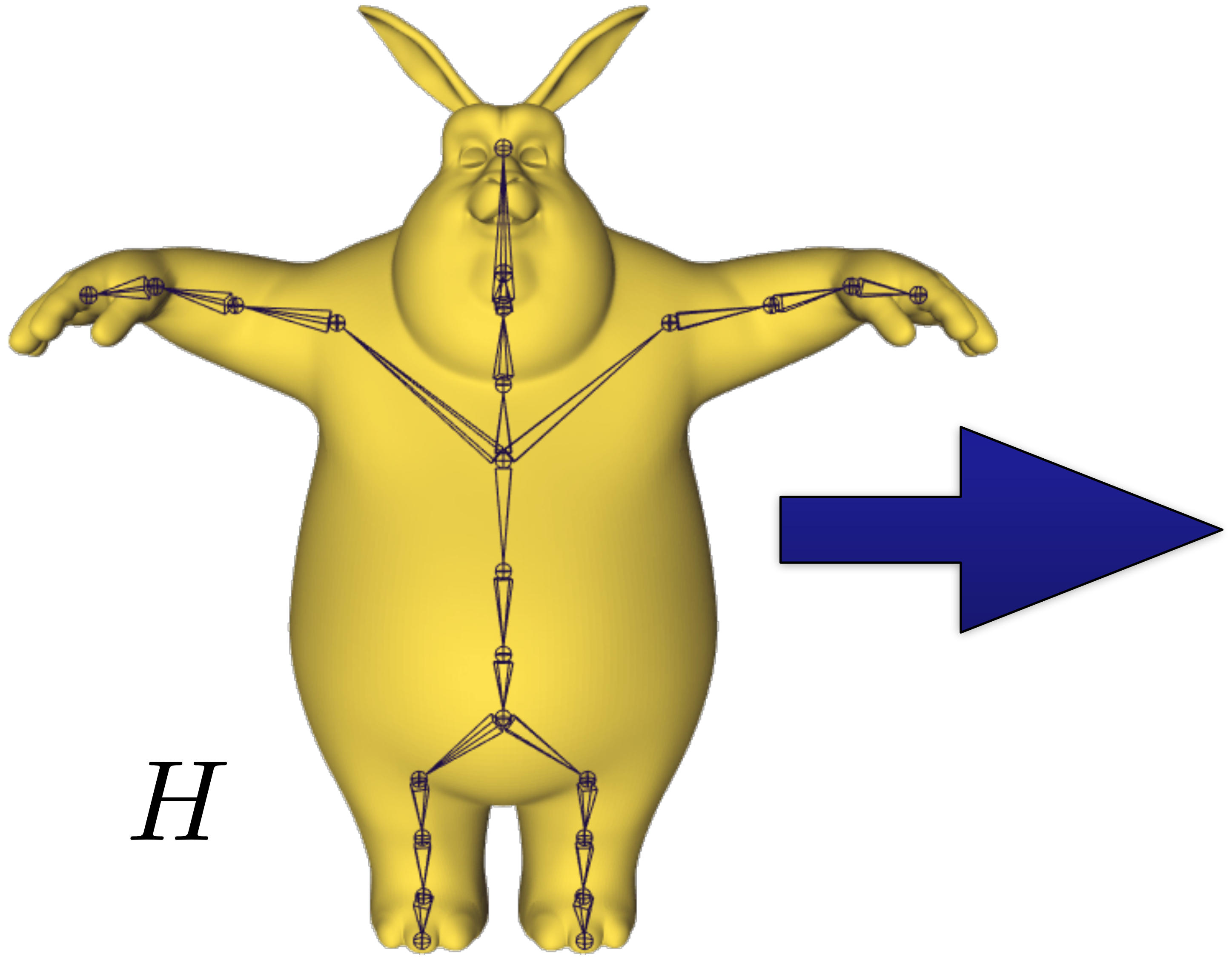
Songrun Liu George Mason University
Jianchao Tan George Mason University
Zhigang Deng University of Houston
Yotam Gingold George Mason University

CraGL
Creativity and Graphics Lab

Linear Blend Skinning (LBS)

$$\mathbf{v}' = \sum_{j \in H} w_j(\mathbf{v}) \mathbf{T}_j \begin{pmatrix} \mathbf{v} \\ 1 \end{pmatrix}$$

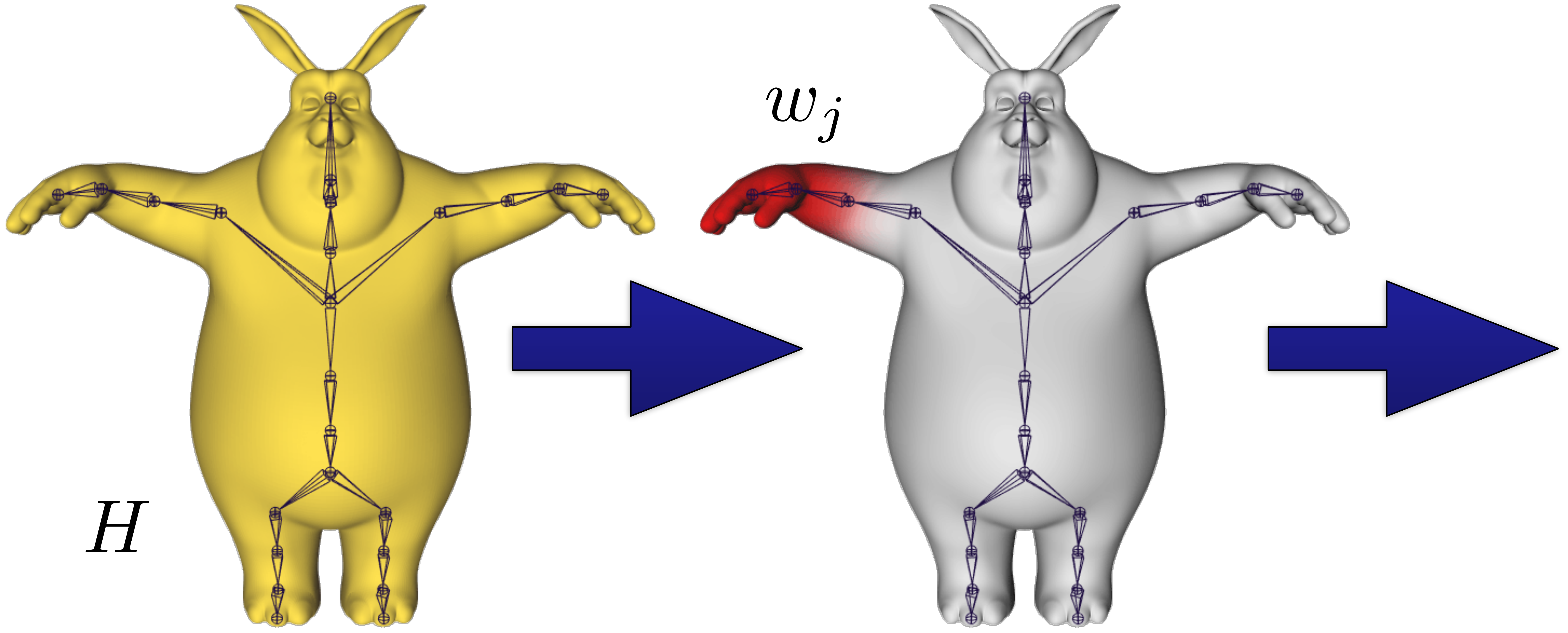
Linear Blend Skinning (LBS)



H

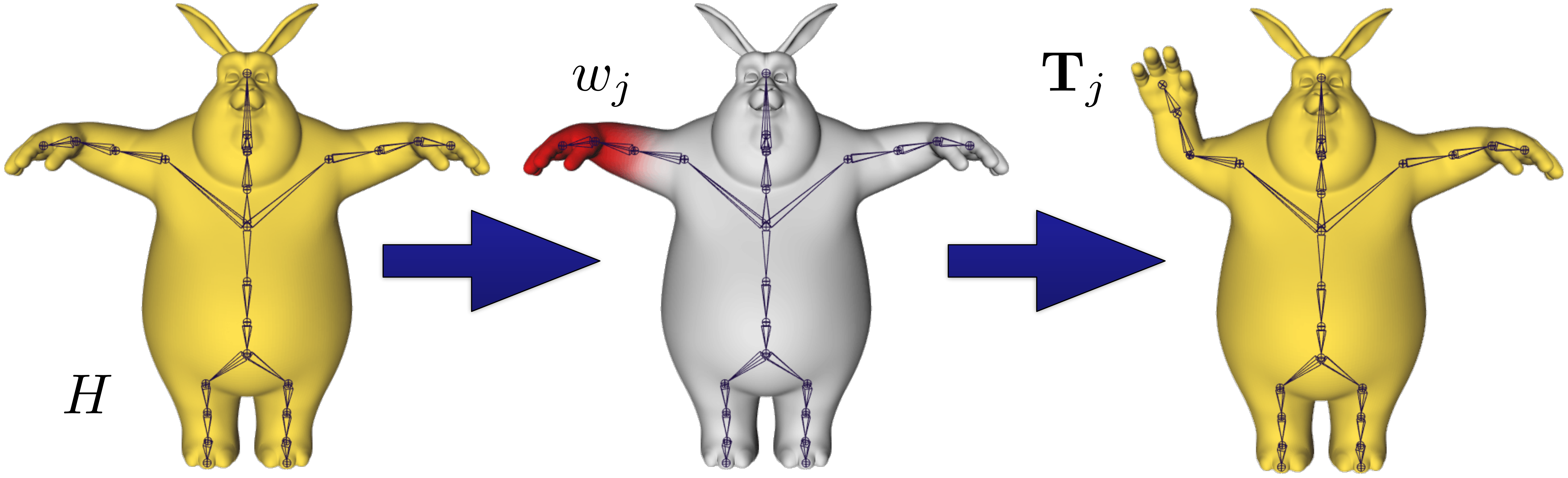
$$\mathbf{v}' = \sum_{j \in \boxed{H}} w_j(\mathbf{v}) \mathbf{T}_j \begin{pmatrix} \mathbf{v} \\ 1 \end{pmatrix}$$

Linear Blend Skinning (LBS)



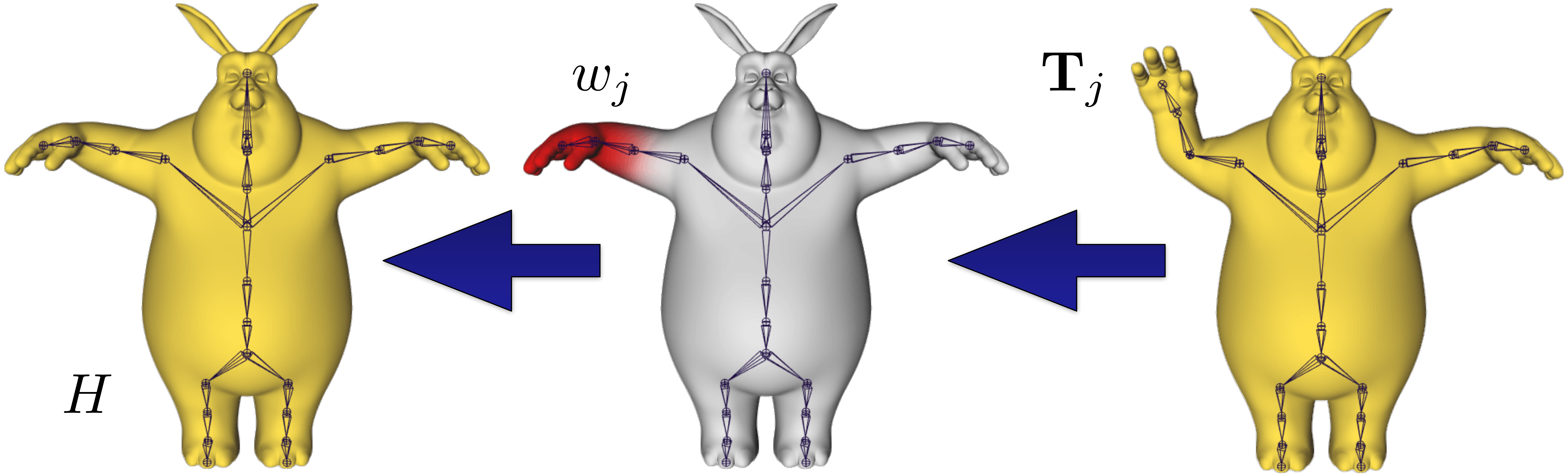
$$\mathbf{v}' = \sum_{j \in H} \boxed{w_j(\mathbf{v})} \mathbf{T}_j \begin{pmatrix} \mathbf{v} \\ 1 \end{pmatrix}$$

Linear Blend Skinning (LBS)



$$\mathbf{v}' = \sum_{j \in H} w_j(\mathbf{v}) \boxed{\mathbf{T}_j} \begin{pmatrix} \mathbf{v} \\ 1 \end{pmatrix}$$

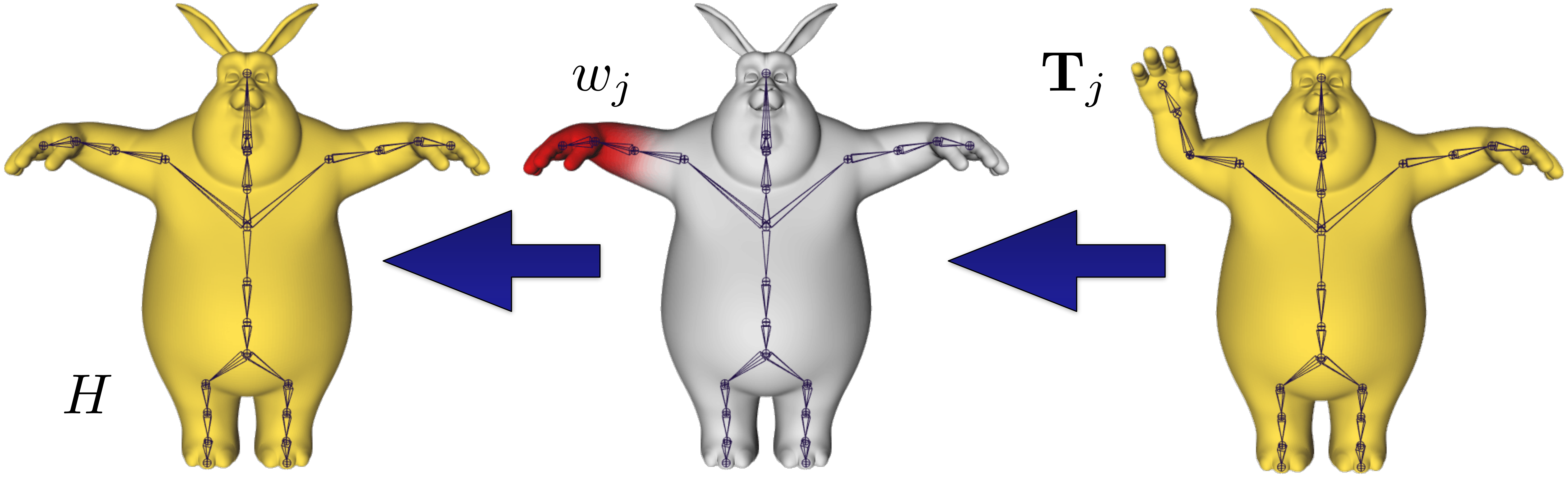
Inverse Linear Blend Skinning



$$\mathbf{v}' = \sum_{j \in H} w_j(\mathbf{v}) \mathbf{T}_j \begin{pmatrix} \mathbf{v} \\ 1 \end{pmatrix}$$

?

Inverse Linear Blend Skinning

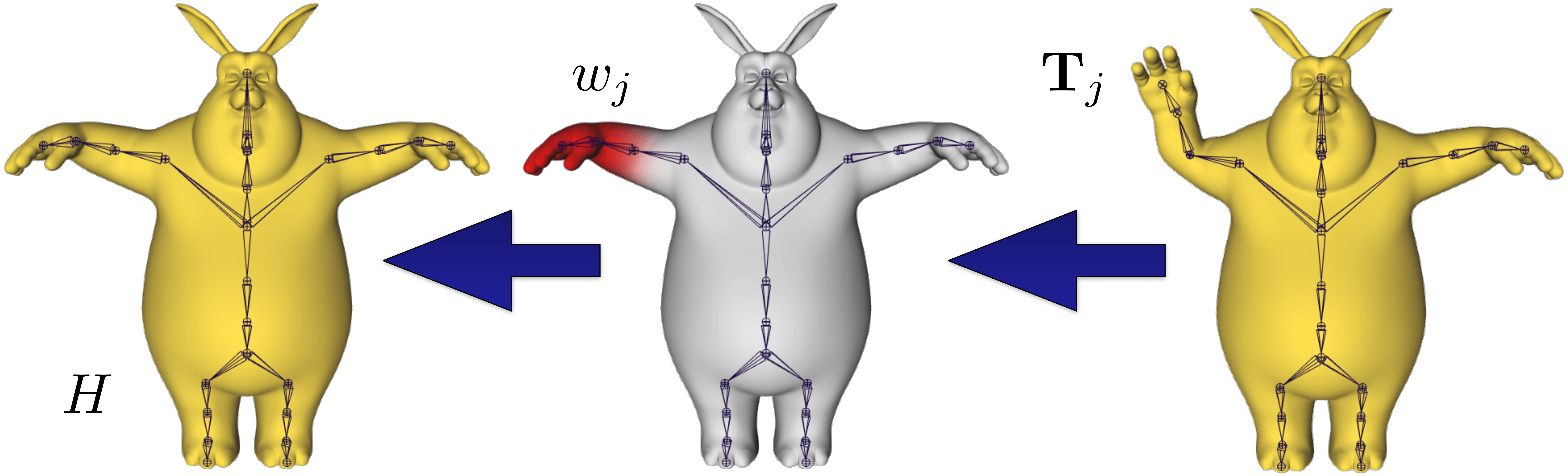


$$\min_{w,R,\mathbf{t},\mathbf{v}} \sum_{p=1}^{\#\text{poses}} \sum_{i=1}^{n} \left\| \mathbf{v}'_{p,i} - \sum_{j=1}^{h} w_{i,j} T_{p,j} \mathbf{v}_i \right\|^2$$

subject to:

$$w_{i,j} \geq 0 \qquad \text{and} \qquad \sum_{j=1}^{h} w_{i,j} = 1$$

Inverse Linear Blend Skinning



$$\min_{w,R,\mathbf{t},\mathbf{v}} \sum_{p=1}^{\#\text{poses}} \sum_{i=1}^{n} \left\| \mathbf{v}'_{p,i} - \sum_{j=1}^{h} w_{i,j} T_{p,j} \mathbf{v}_i \right\|^2$$

subject to:

$$w_{i,j} \geq 0 \quad \text{and} \quad \sum_{j=1}^{h} w_{i,j} = 1$$

Previous Work:

[James and Twigg 2005]
[Schaefer and Yuksel 2007]
[De Aguiar et al. 2008]
[Hasler et al. 2010]
[Kavan et al. 2010]
[Le and Deng 2012, 2013, 2014]

Inverse LBS is a problem in high-dimensions

Inverse LBS is a problem in high-dimensions

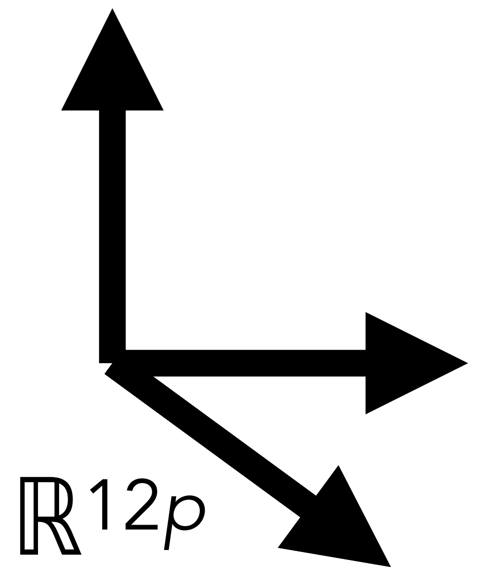
- Transformation matrices are affine: $\mathbb{R}^{12}$

Inverse LBS is a problem in high-dimensions

- Transformation matrices are affine: $\mathbb{R}^{12}$
- Handles have transformations across all animation frames or poses: $\mathbb{R}^{12p}$

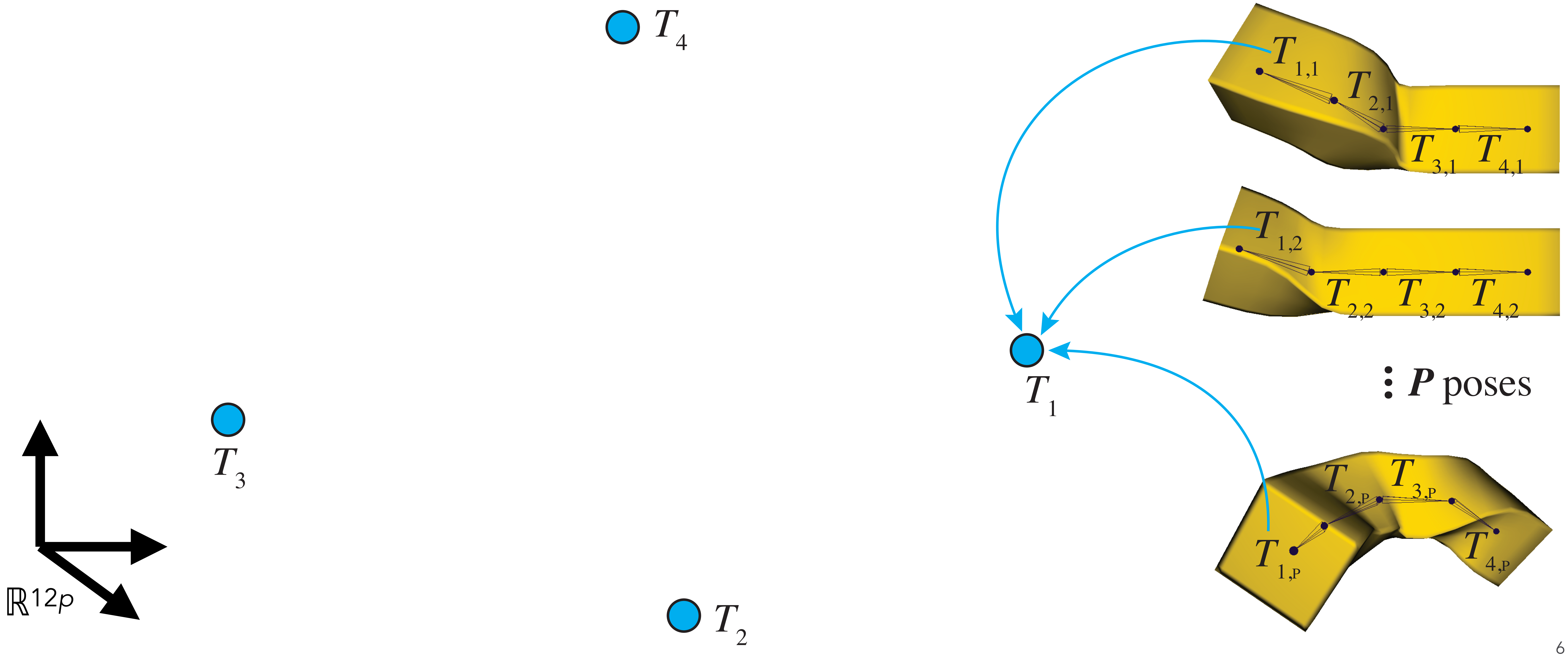
Inverse LBS is a problem in high-dimensions

- Transformation matrices are affine: $\mathbb{R}^{12}$
- Handles have transformations across all animation frames or poses: $\mathbb{R}^{12p}$



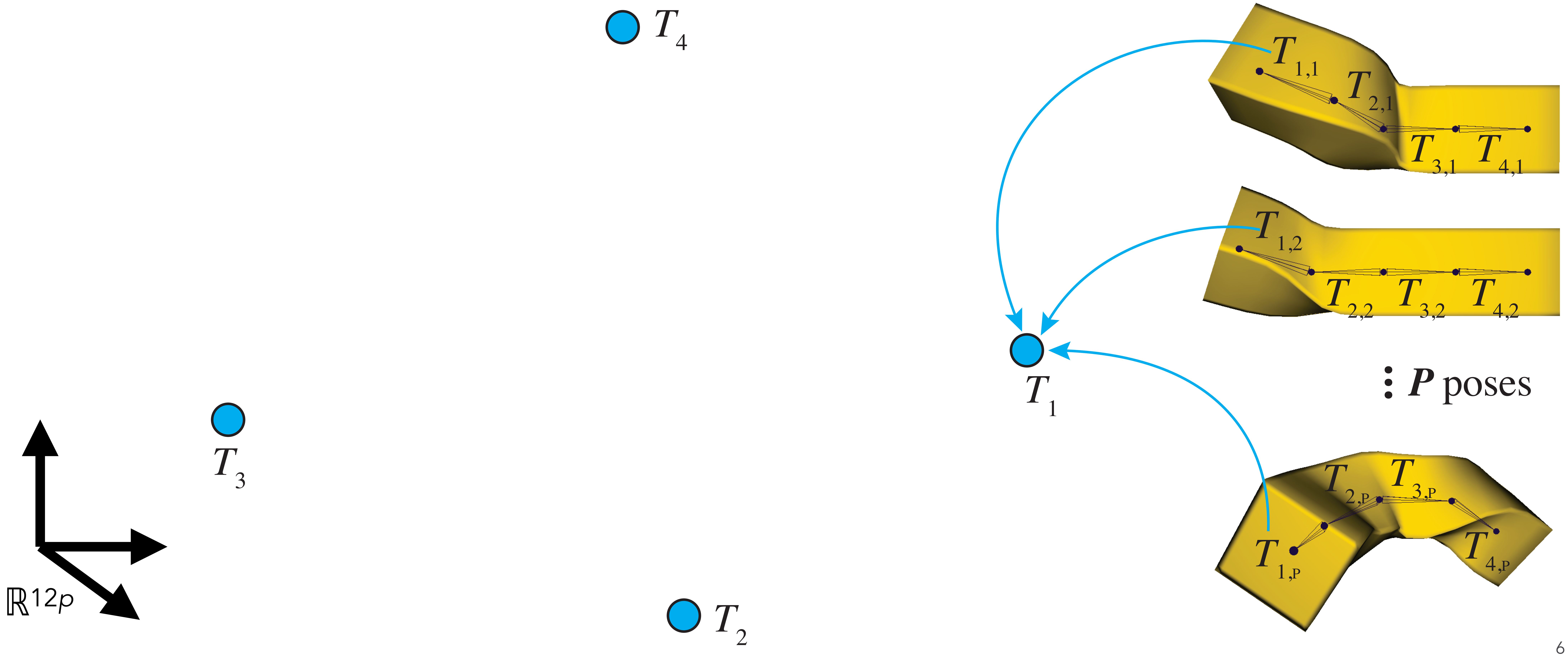
Inverse LBS is a problem in high-dimensions

- Transformation matrices are affine: $\mathbb{R}^{12}$
- Handles have transformations across all animation frames or poses: $\mathbb{R}^{12p}$



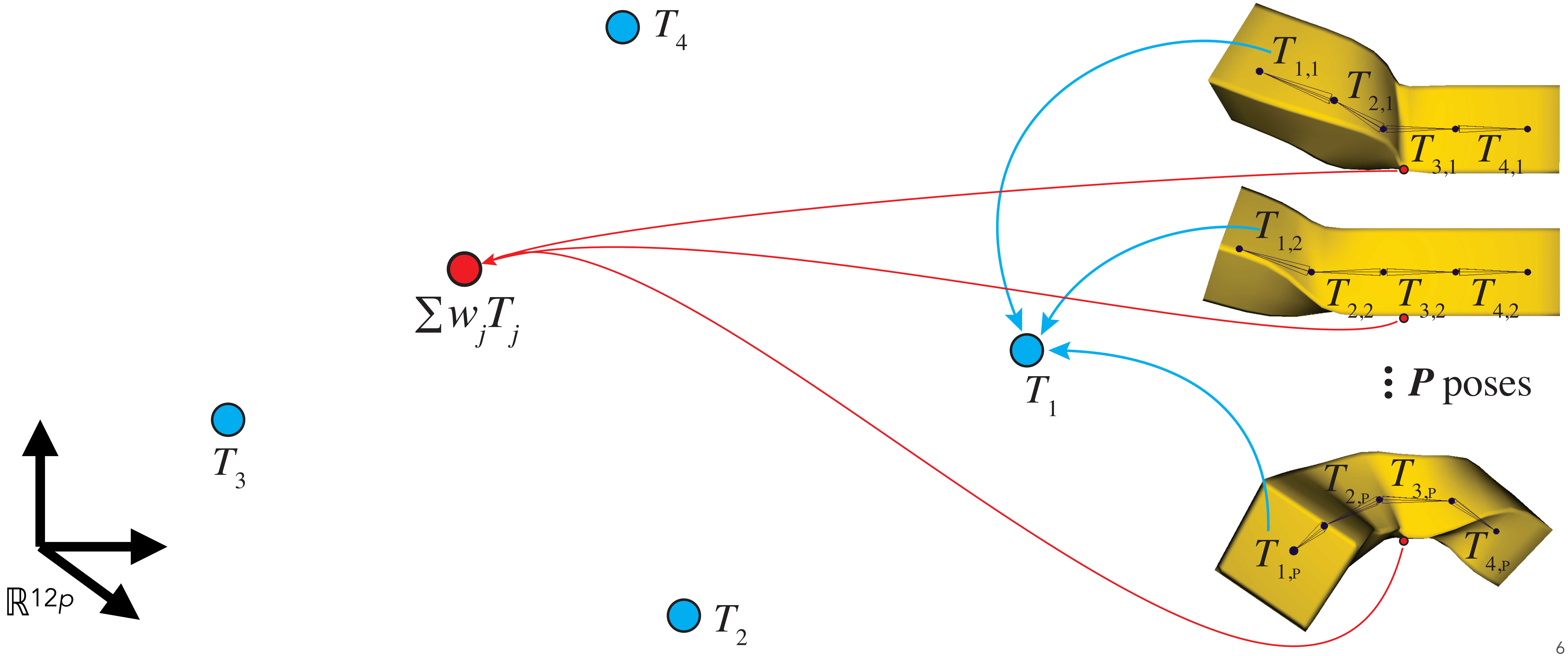
Inverse LBS is a problem in high-dimensions

- Transformation matrices are affine: $\mathbb{R}^{12}$
- Handles have transformations across all animation frames or poses: $\mathbb{R}^{12p}$
- LBS takes weighted averages of these transformations



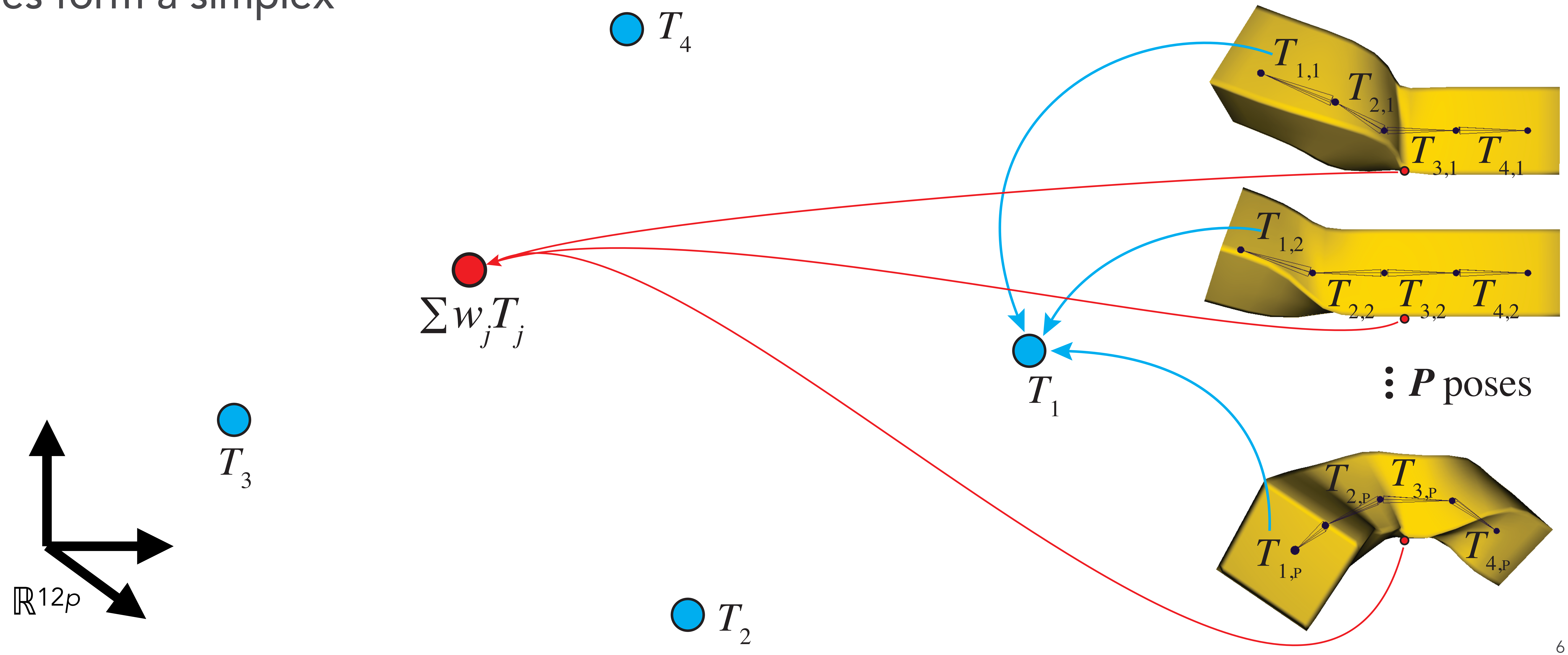
Inverse LBS is a problem in high-dimensions

- Transformation matrices are affine: $\mathbb{R}^{12}$
- Handles have transformations across all animation frames or poses: $\mathbb{R}^{12p}$
- LBS takes weighted averages of these transformations



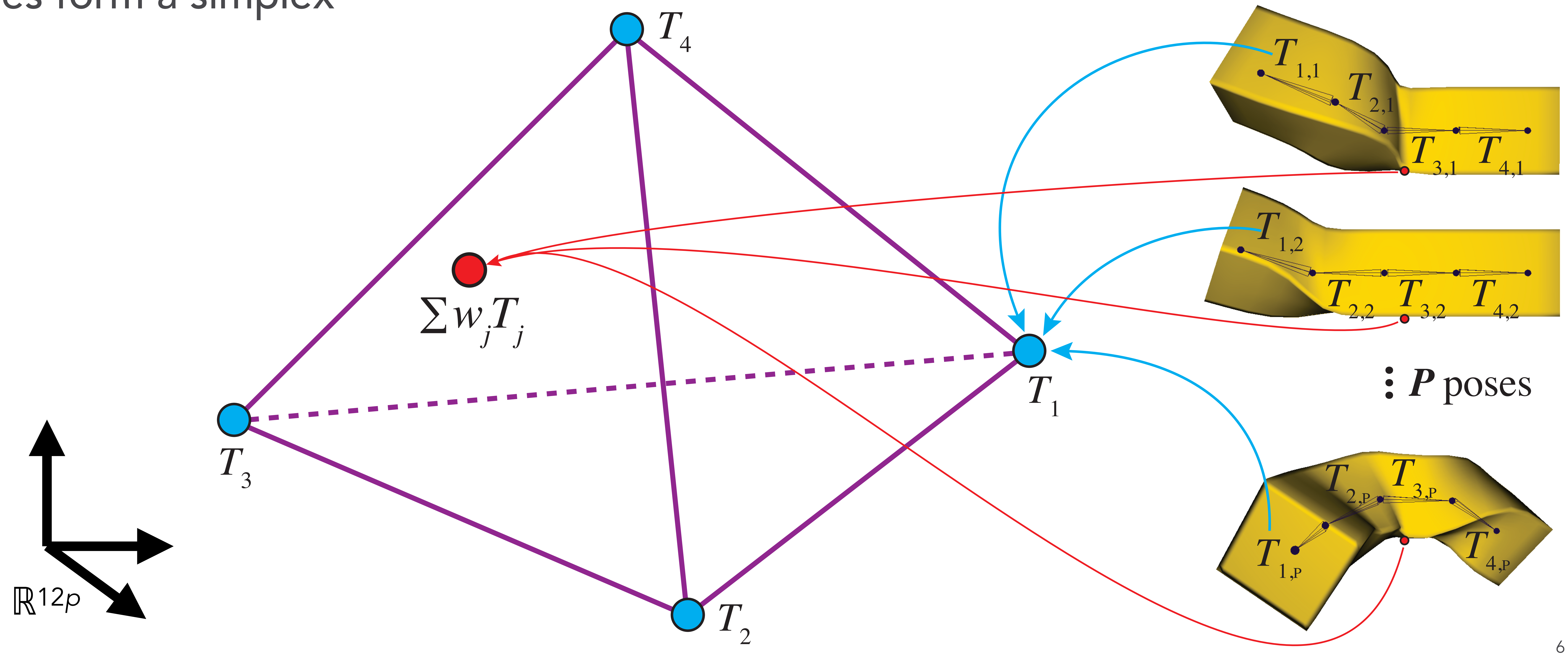
Inverse LBS is a problem in high-dimensions

- Transformation matrices are affine: $\mathbb{R}^{12}$
- Handles have transformations across all animation frames or poses: $\mathbb{R}^{12p}$
- LBS takes weighted averages of these transformations
- The handles form a simplex



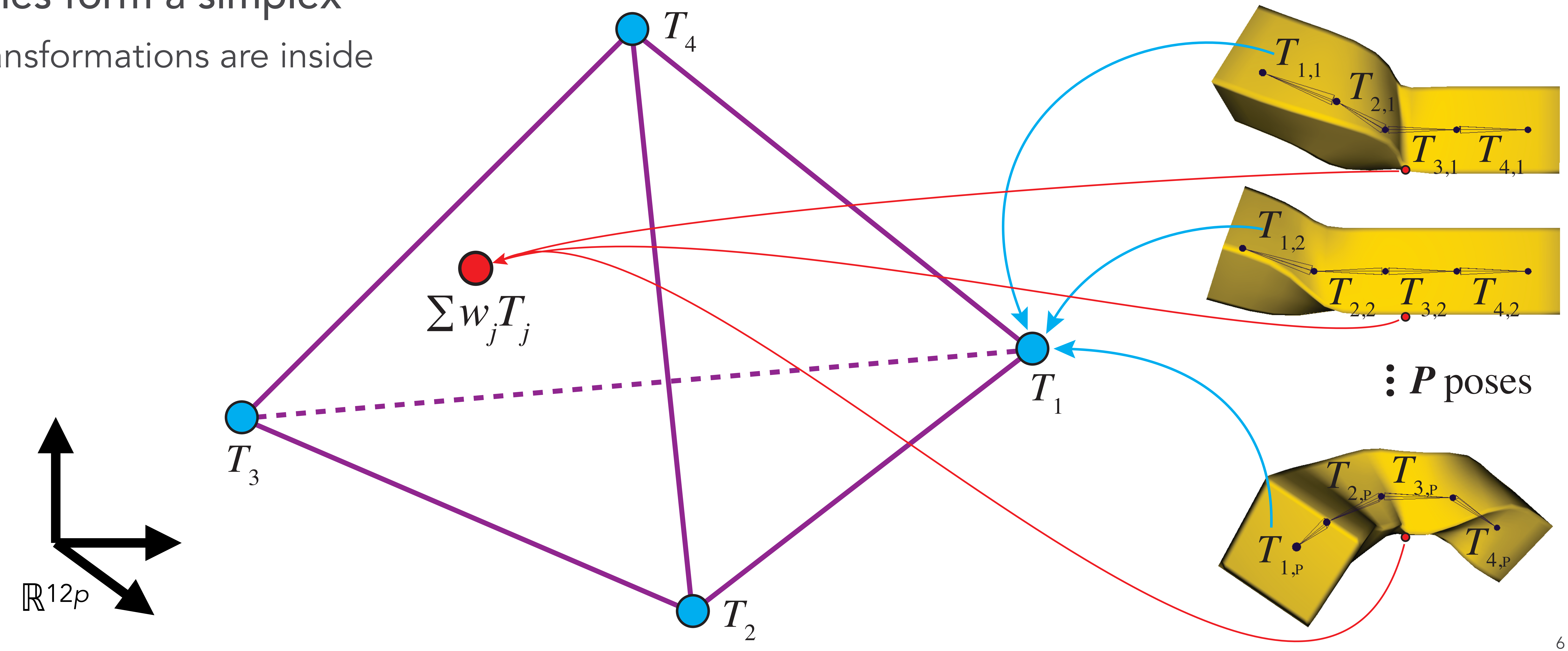
Inverse LBS is a problem in high-dimensions

- Transformation matrices are affine: $\mathbb{R}^{12}$
- Handles have transformations across all animation frames or poses: $\mathbb{R}^{12p}$
- LBS takes weighted averages of these transformations
- The handles form a simplex



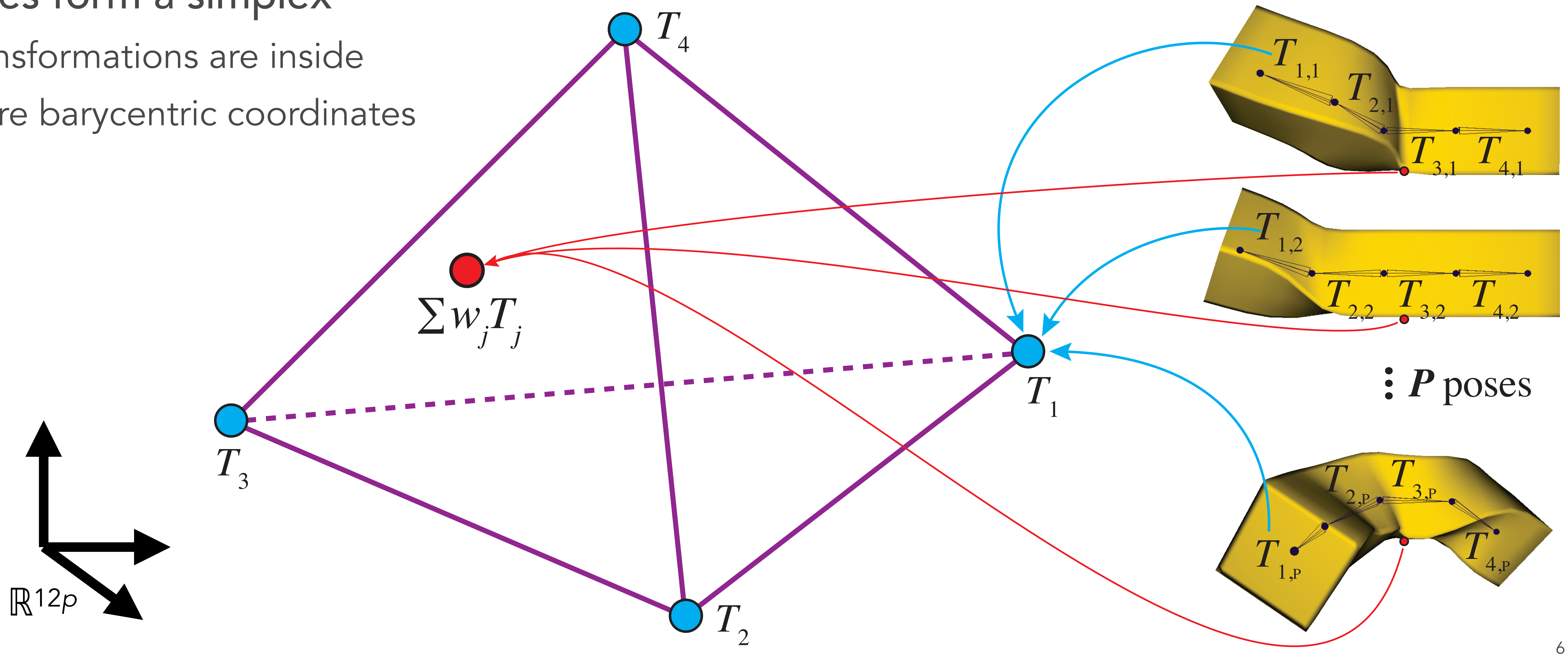
Inverse LBS is a problem in high-dimensions

- Transformation matrices are affine: $\mathbb{R}^{12}$
- Handles have transformations across all animation frames or poses: $\mathbb{R}^{12p}$
- LBS takes weighted averages of these transformations
- The handles form a simplex
  - Vertex transformations are inside



Inverse LBS is a problem in high-dimensions

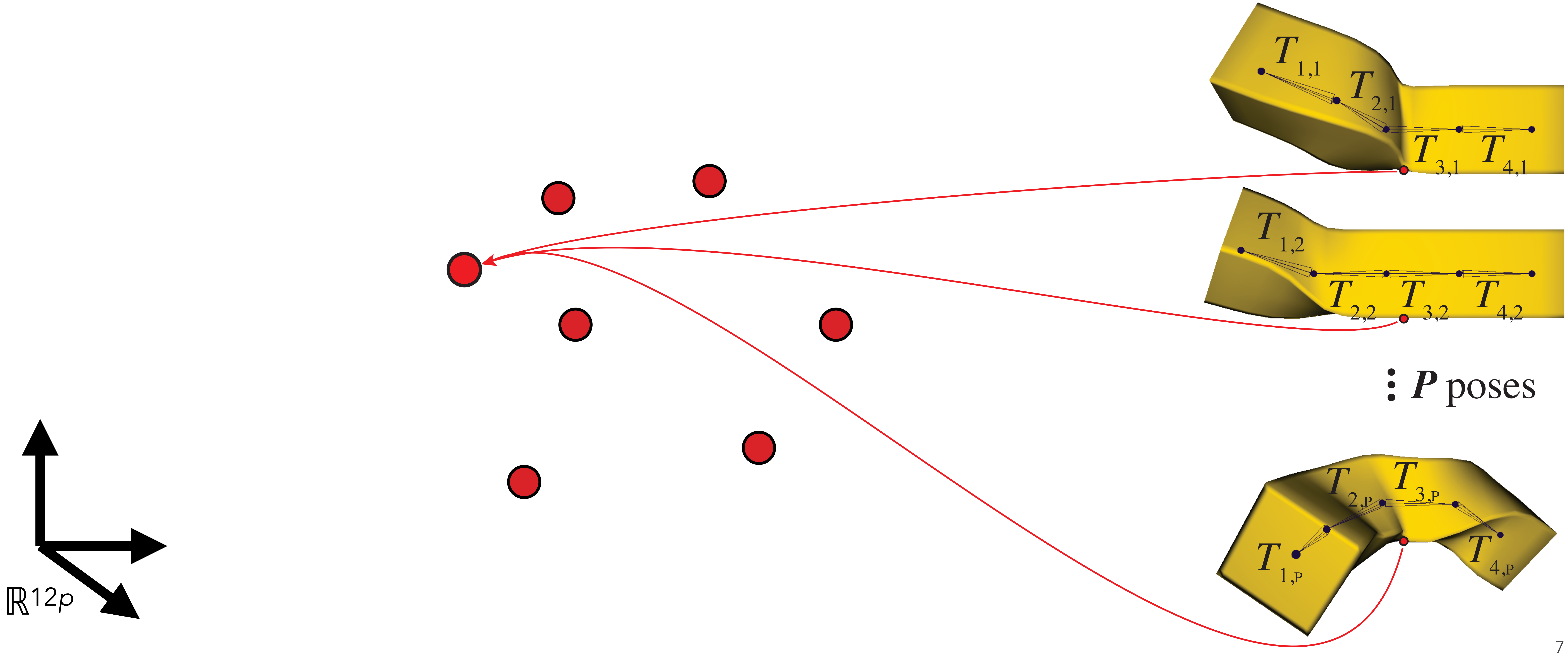
- Transformation matrices are affine: $\mathbb{R}^{12}$
- Handles have transformations across all animation frames or poses: $\mathbb{R}^{12p}$
- LBS takes weighted averages of these transformations
- The handles form a simplex
  - Vertex transformations are inside
  - Weights are barycentric coordinates



Our Approach

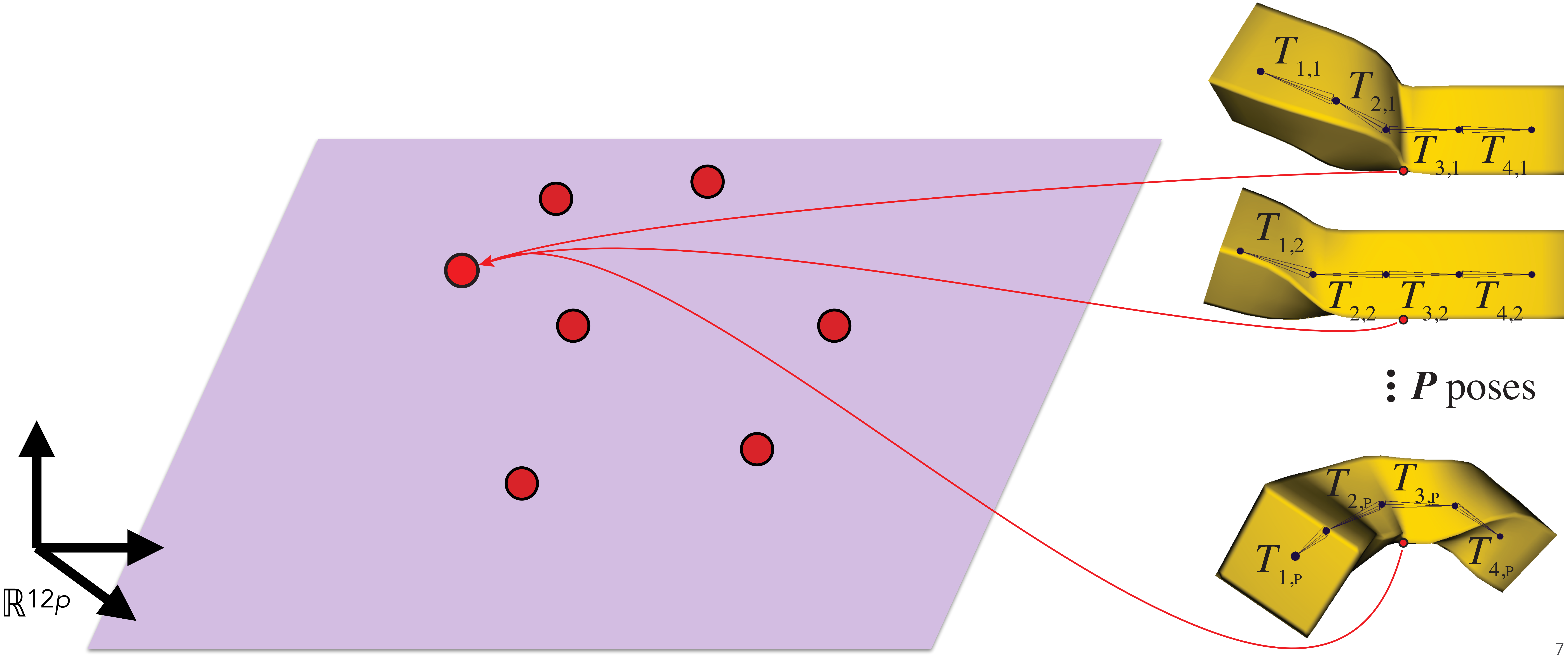
Our Approach

- Step 1: Estimate vertex transformations in $\mathbb{R}^{12p}$



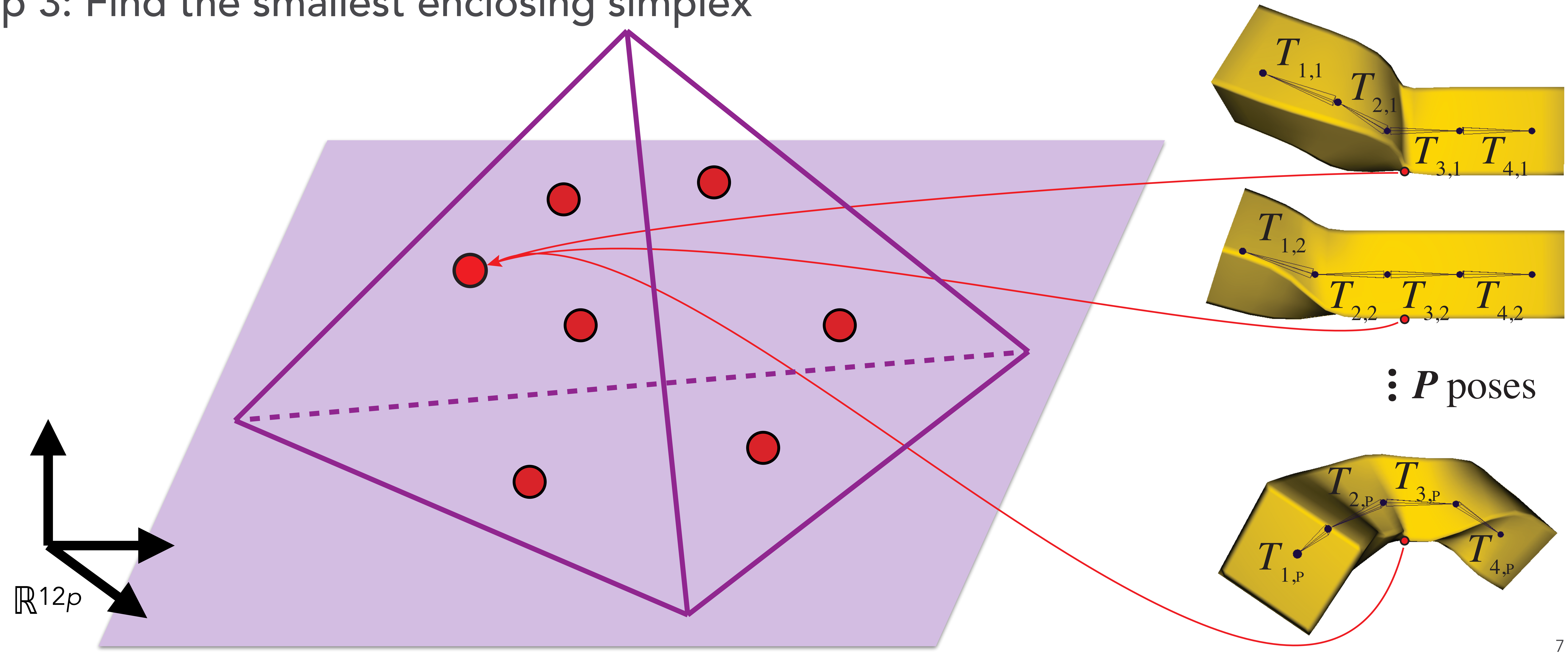
Our Approach

- Step 1: Estimate vertex transformations in $\mathbb{R}^{12p}$
- Step 2: Estimate a #*handles*-dimensional subspace for the vertices



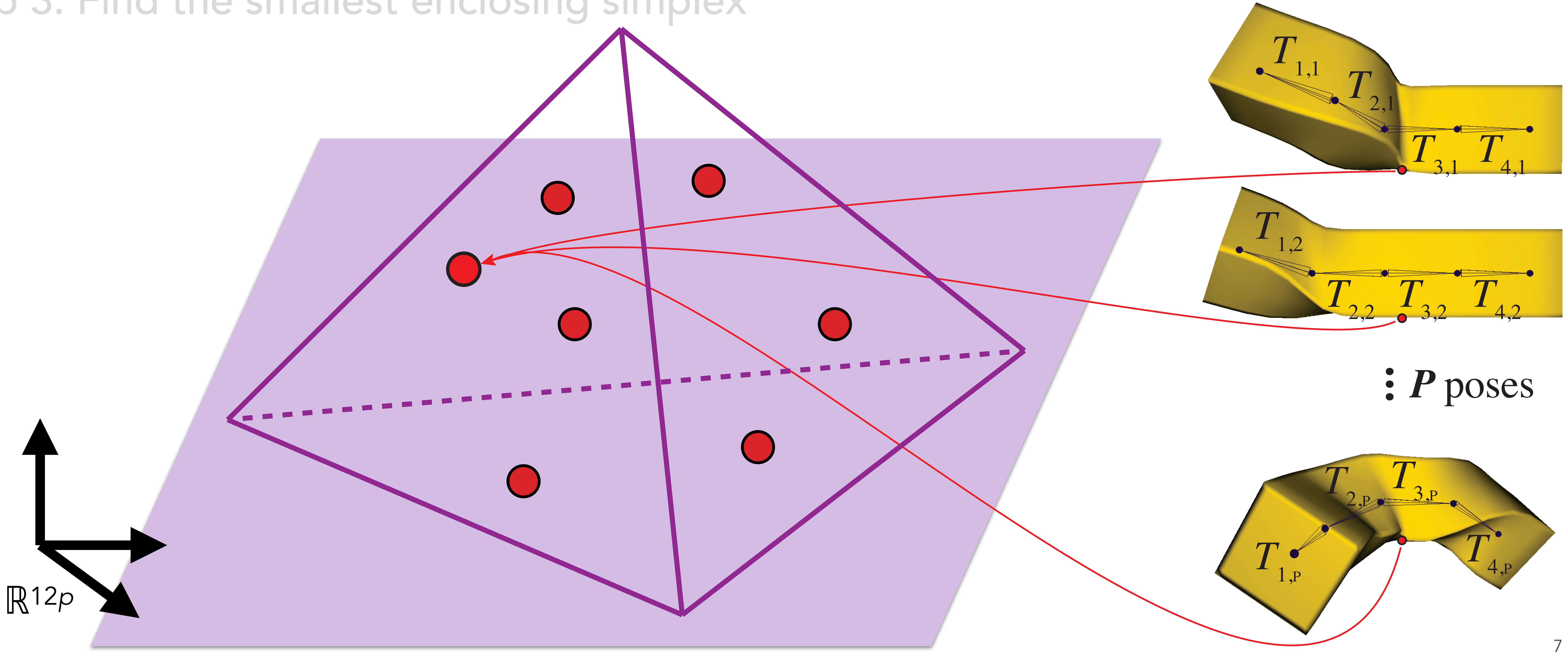
Our Approach

- Step 1: Estimate vertex transformations in $\mathbb{R}^{12p}$
- Step 2: Estimate a #*handles*-dimensional subspace for the vertices
- Step 3: Find the smallest enclosing simplex



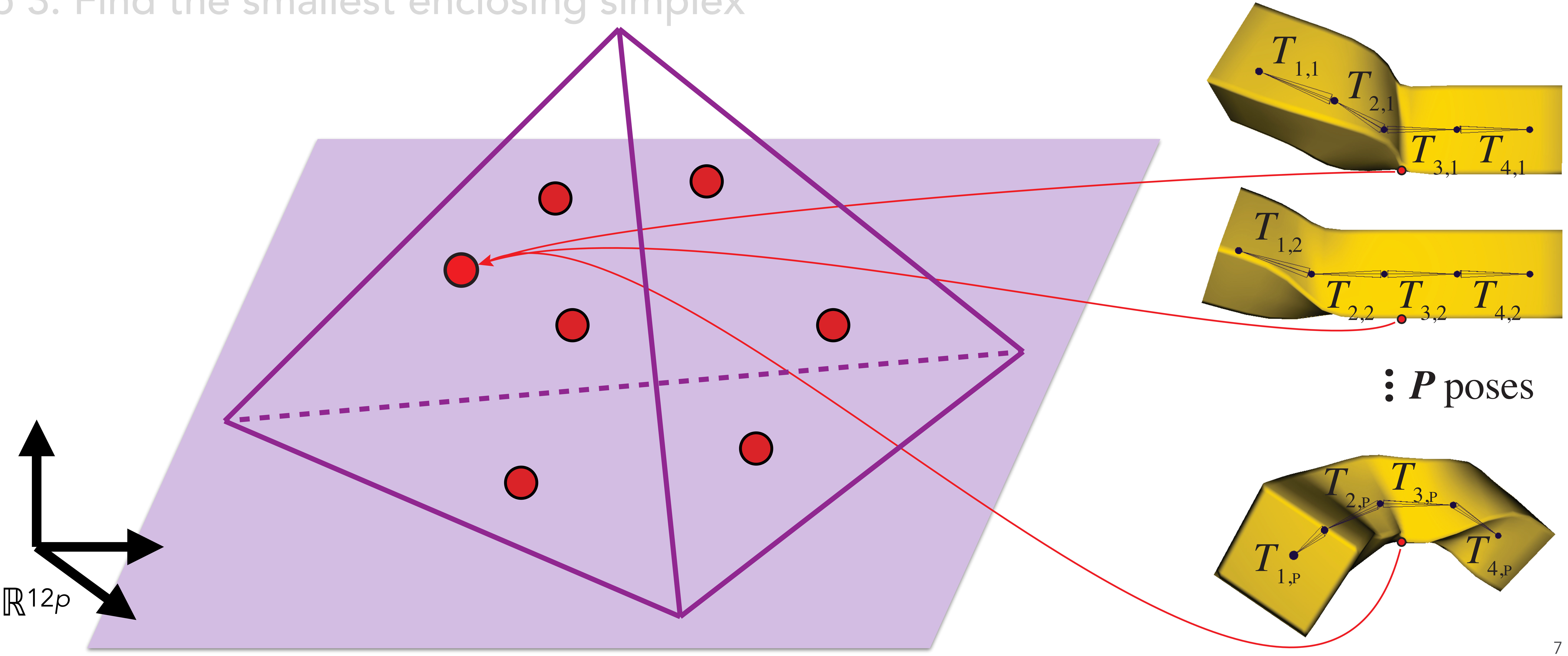
Our Approach

- Step 1: Estimate vertex transformations in $\mathbb{R}^{12p}$
- Step 2: Estimate a #*handles*-dimensional subspace for the vertices
- Step 3: Find the smallest enclosing simplex



Our Approach

- Step 1: Estimate vertex transformations in $\mathbb{R}^{12p}$
- Step 2: Estimate a #*handles*-dimensional subspace for the vertices
- Step 3: Find the smallest enclosing simplex

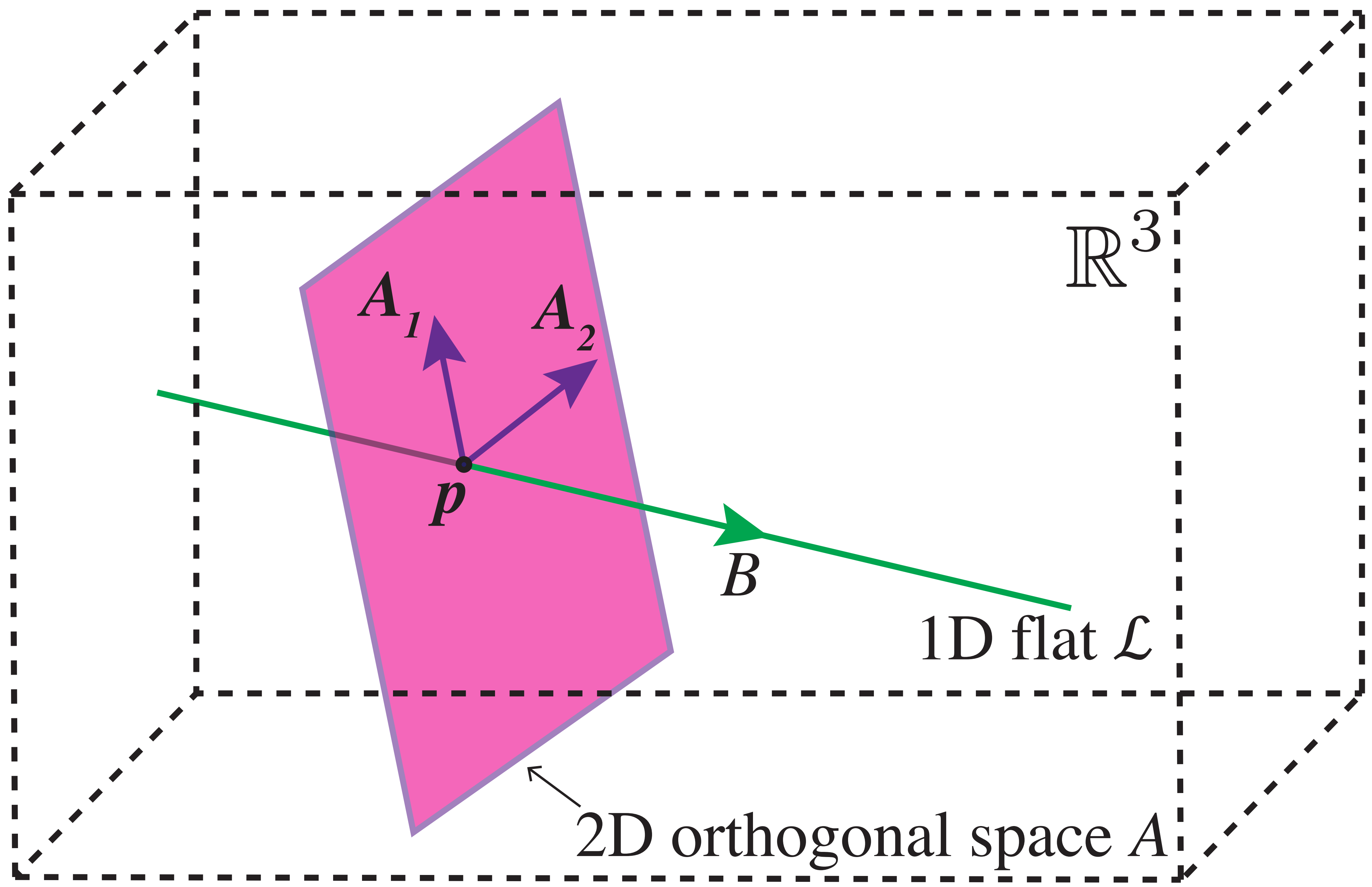


Step 1: Estimate vertex positions in $\mathbb{R}^{12p}$

- For each pose, we know the vertex's rest and deformed position. This constrains possible handle transformations to an affine subspace or *flat* in $\mathbb{R}^{9p}$

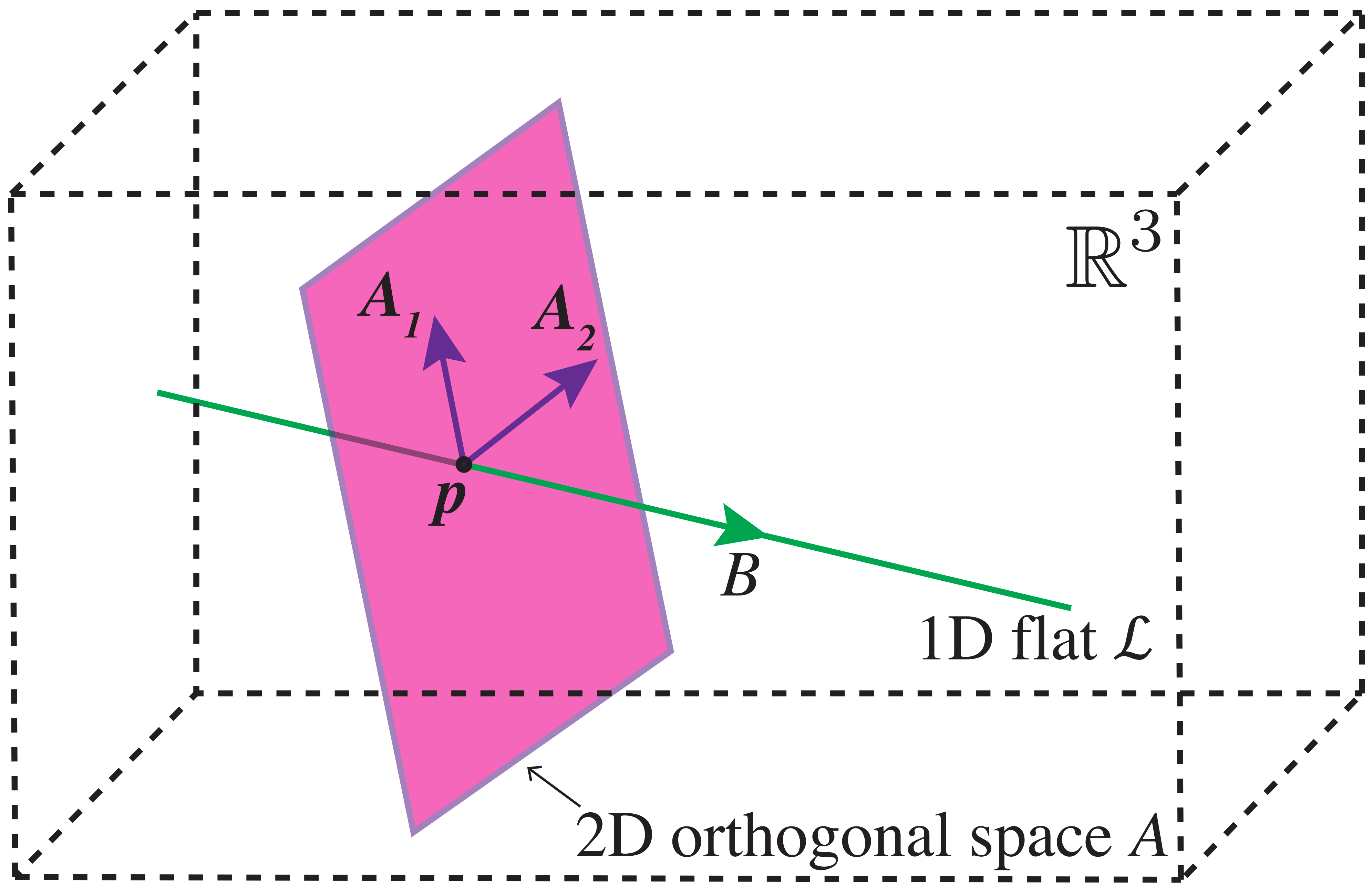
$$\bar{V}_i \mathbf{x} = \begin{bmatrix} \mathbf{v}'_{1,i} \\ \vdots \\ \mathbf{v}'_{p,i} \end{bmatrix} = \mathbf{v}'_i$$

Flats…



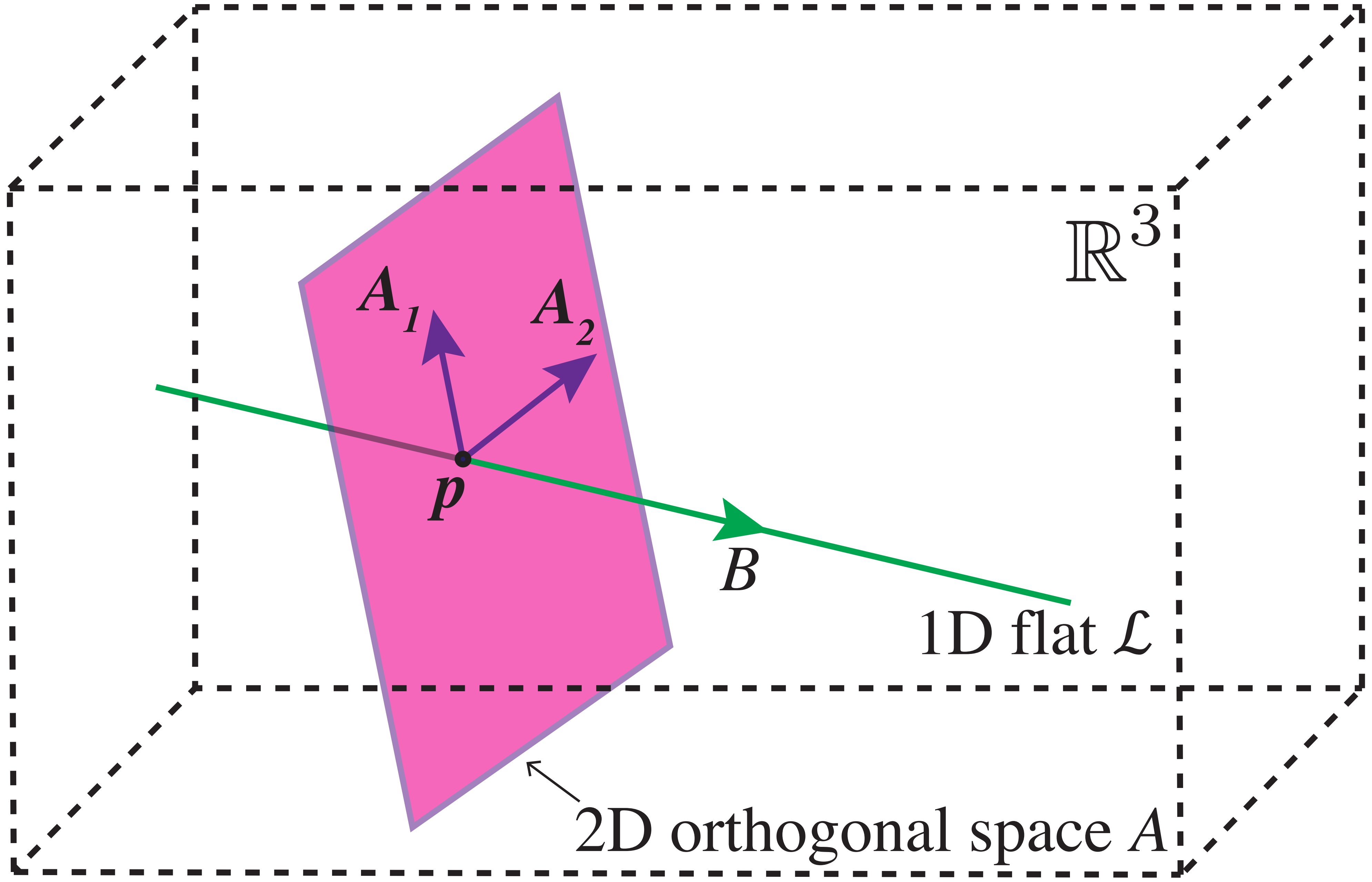
Flats...

- ... generalize a line or plane (a linear subspace offset from the origin) to higher dimensions



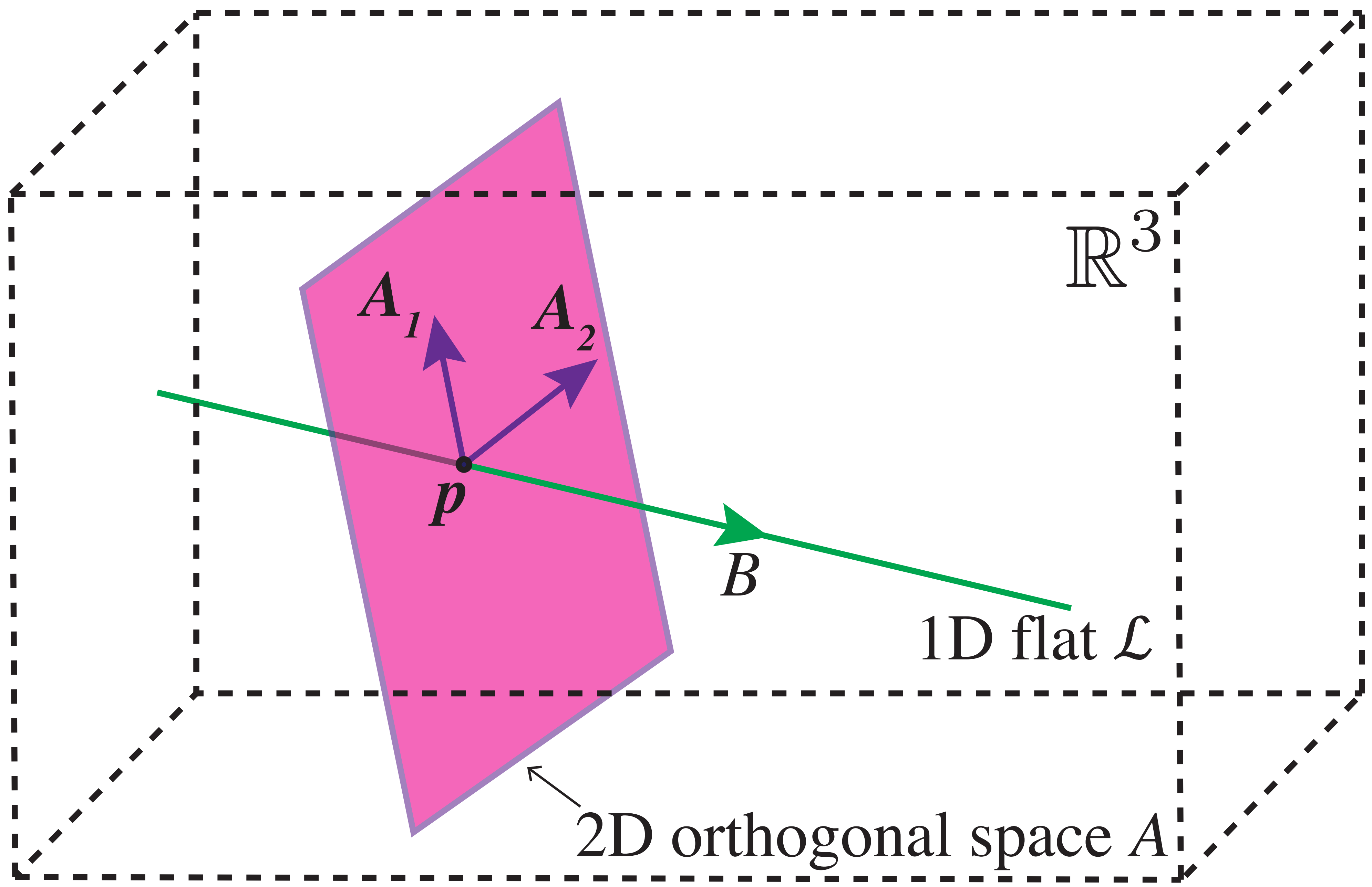
Flats...

- ... generalize a line or plane (a linear subspace offset from the origin) to higher dimensions
- ... can be defined explicitly: $\mathscr{L} = \{\mathbf{p} + B\mathbf{z}\}$



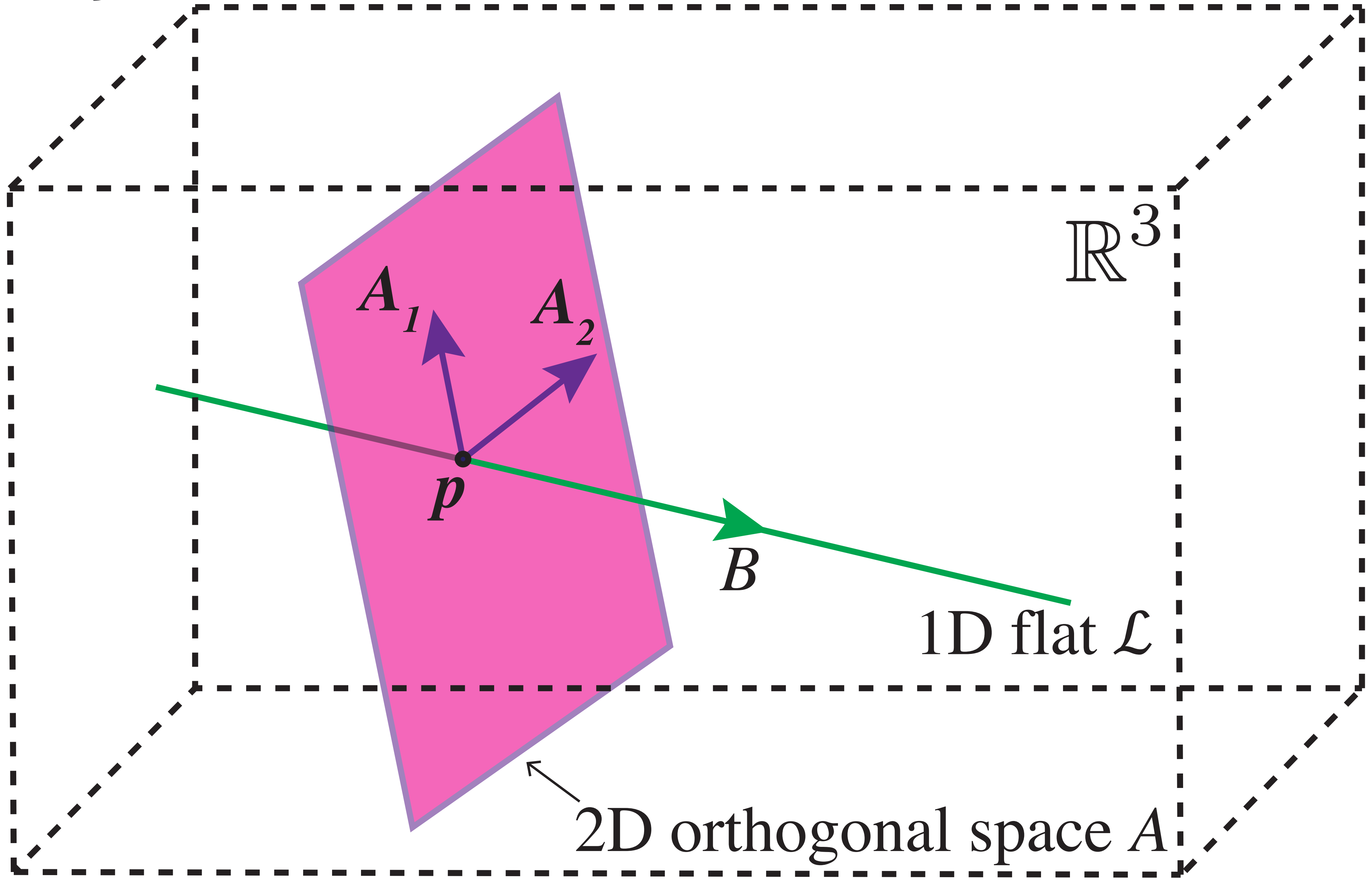
Flats…

- … generalize a line or plane (a linear subspace offset from the origin) to higher dimensions
- … can be defined explicitly: $\mathscr{L} = \{\mathbf{p} + B\mathbf{z}\}$
- … can be defined as weighted average: $\mathscr{L} = \{F\mathbf{w}\}$



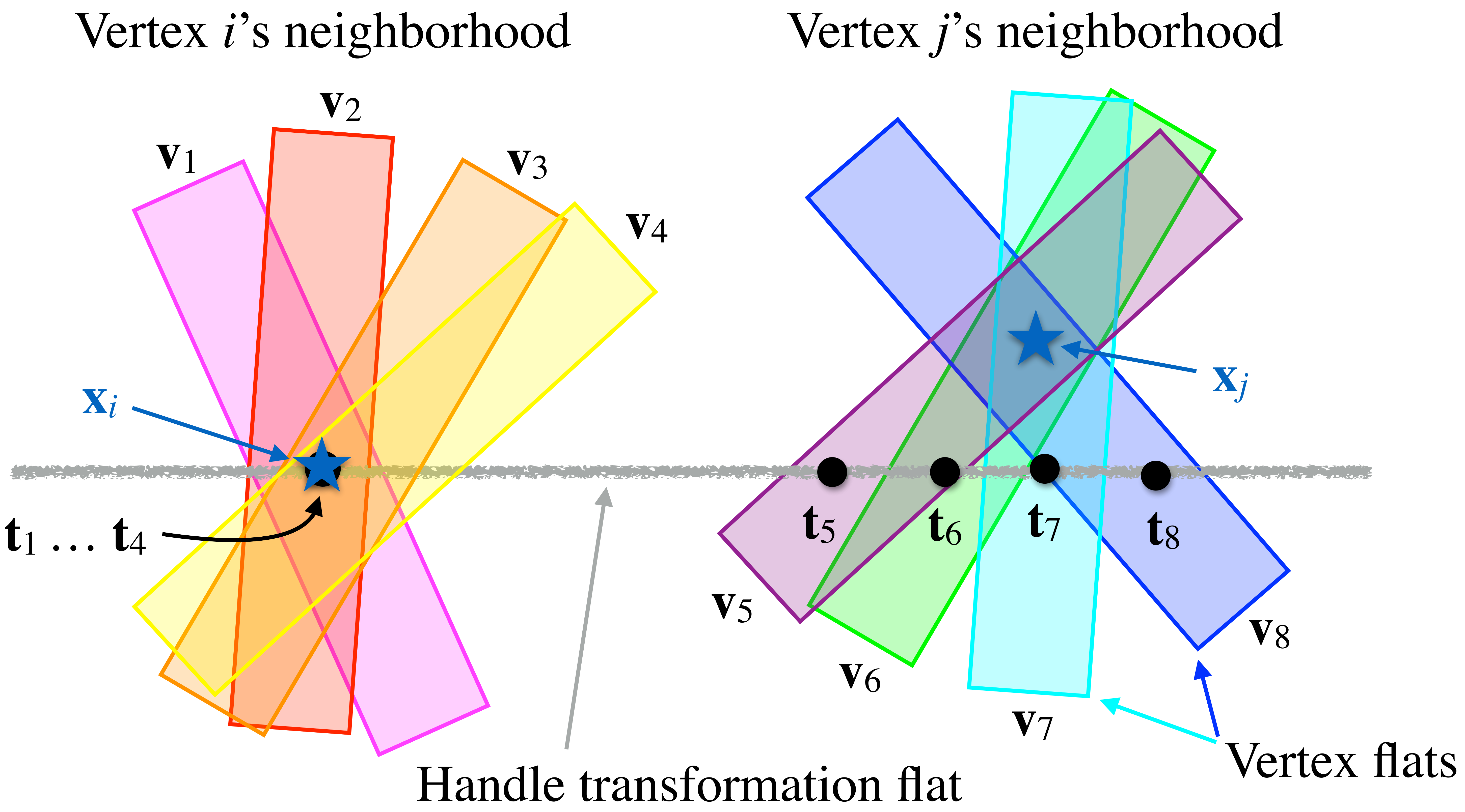
Flats...

- ... generalize a line or plane (a linear subspace offset from the origin) to higher dimensions
- ... can be defined explicitly: $\mathscr{L} = \{\mathbf{p} + B\mathbf{z}\}$
- ... can be defined as weighted average: $\mathscr{L} = \{F\mathbf{w}\}$
- ... can be defined implicitly: $\mathscr{L} = \{\mathbf{x} \in \mathbb{R}^n \mid A\mathbf{x} = \mathbf{a}\}$

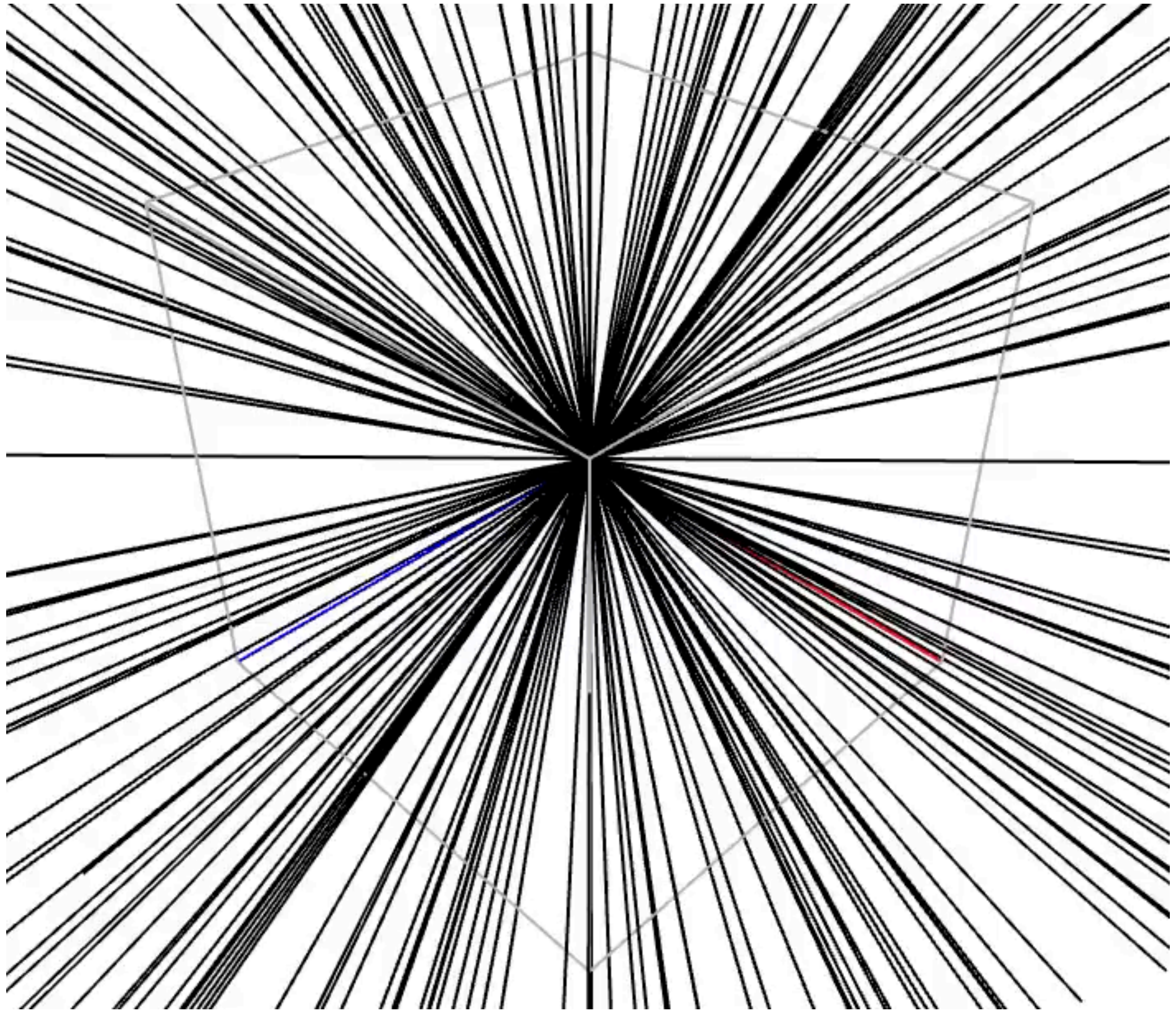


Step 2: Estimate a *handle* subspace close to the vertices

- We want a (#*handles-1*)-dimensional flat that intersects or is as close as possible to all individual vertices' flats.

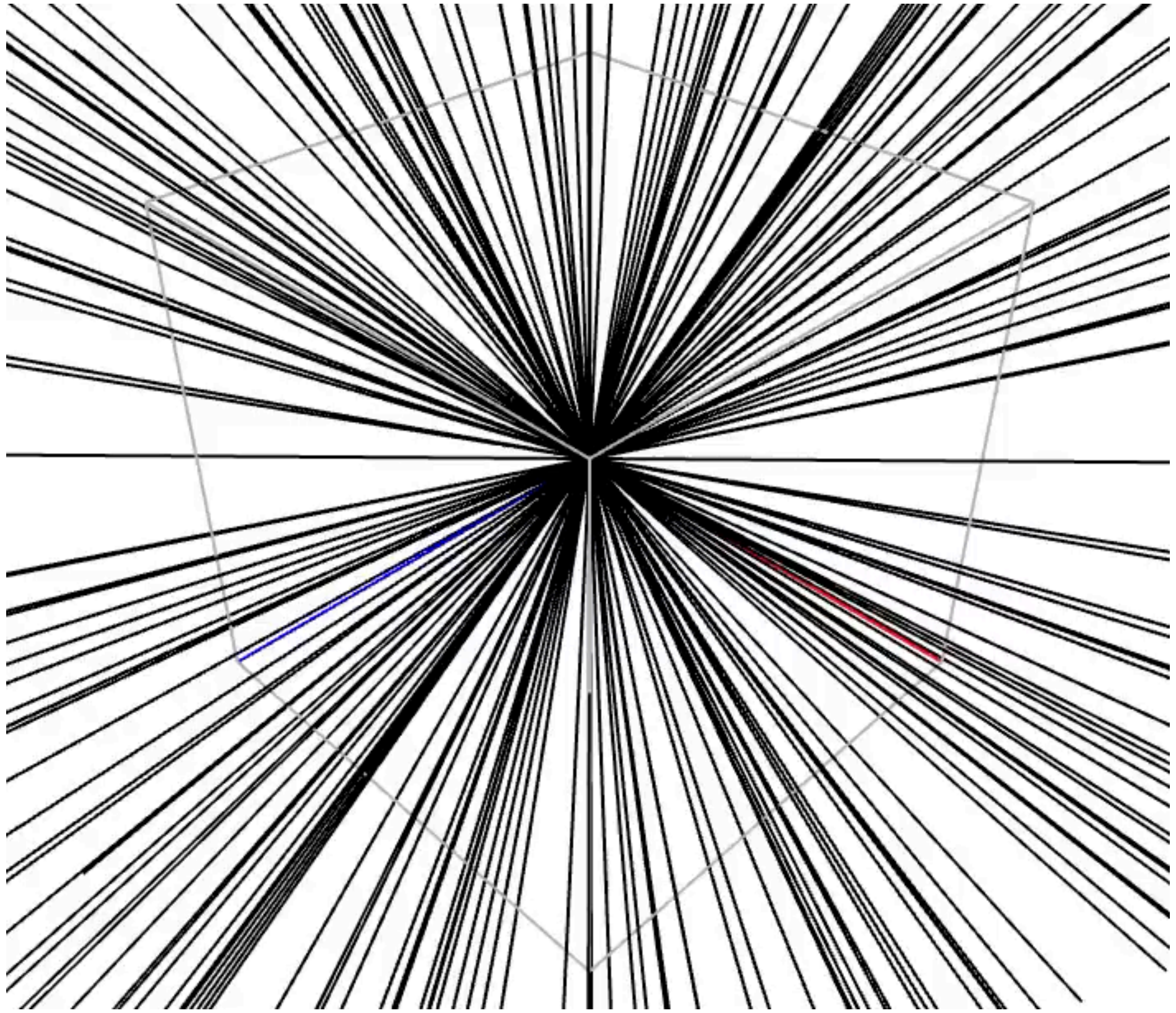


The Closest Flat Problem is Hard



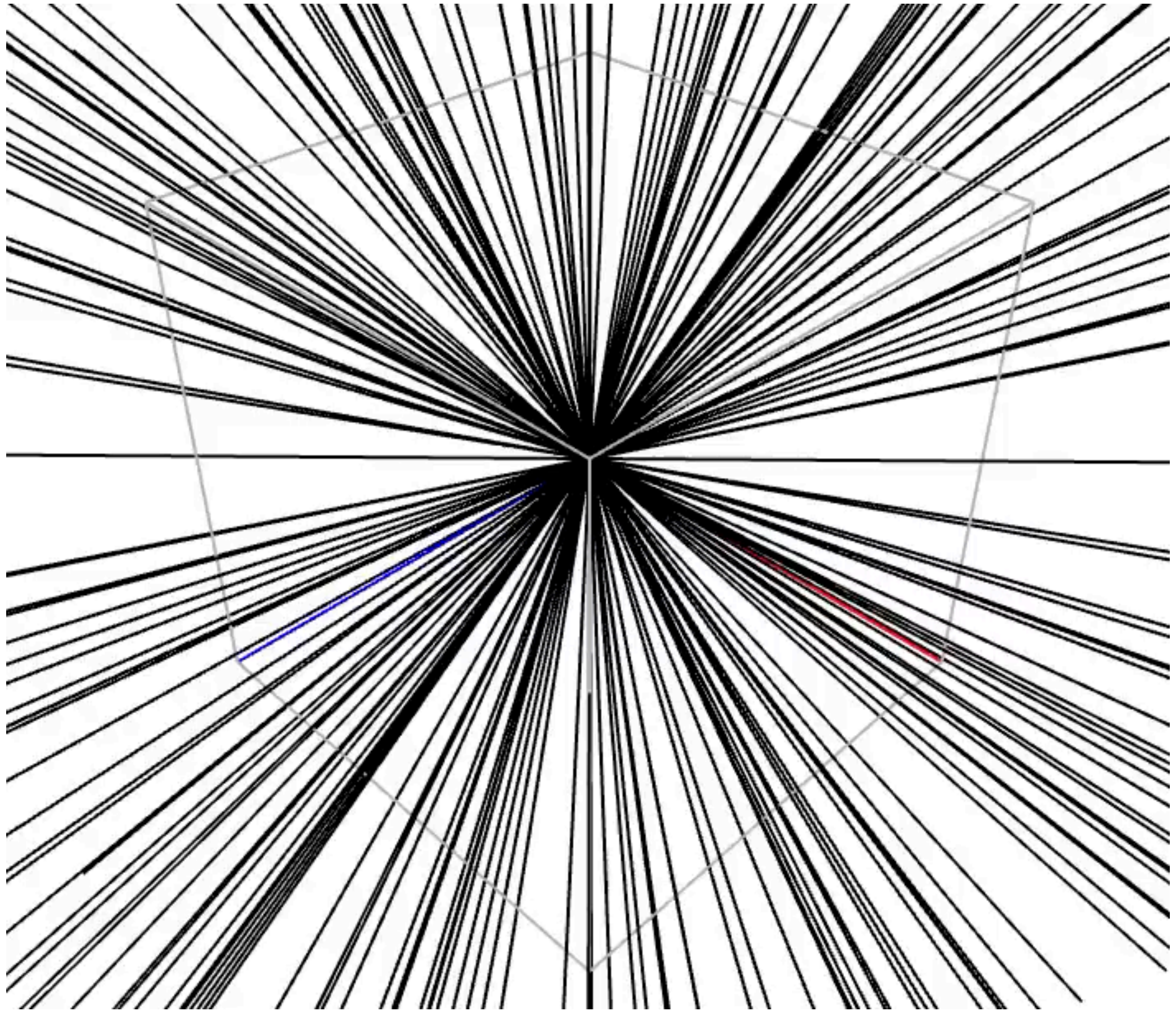
The Closest Flat Problem is Hard

- It's not convex. How hard is it?



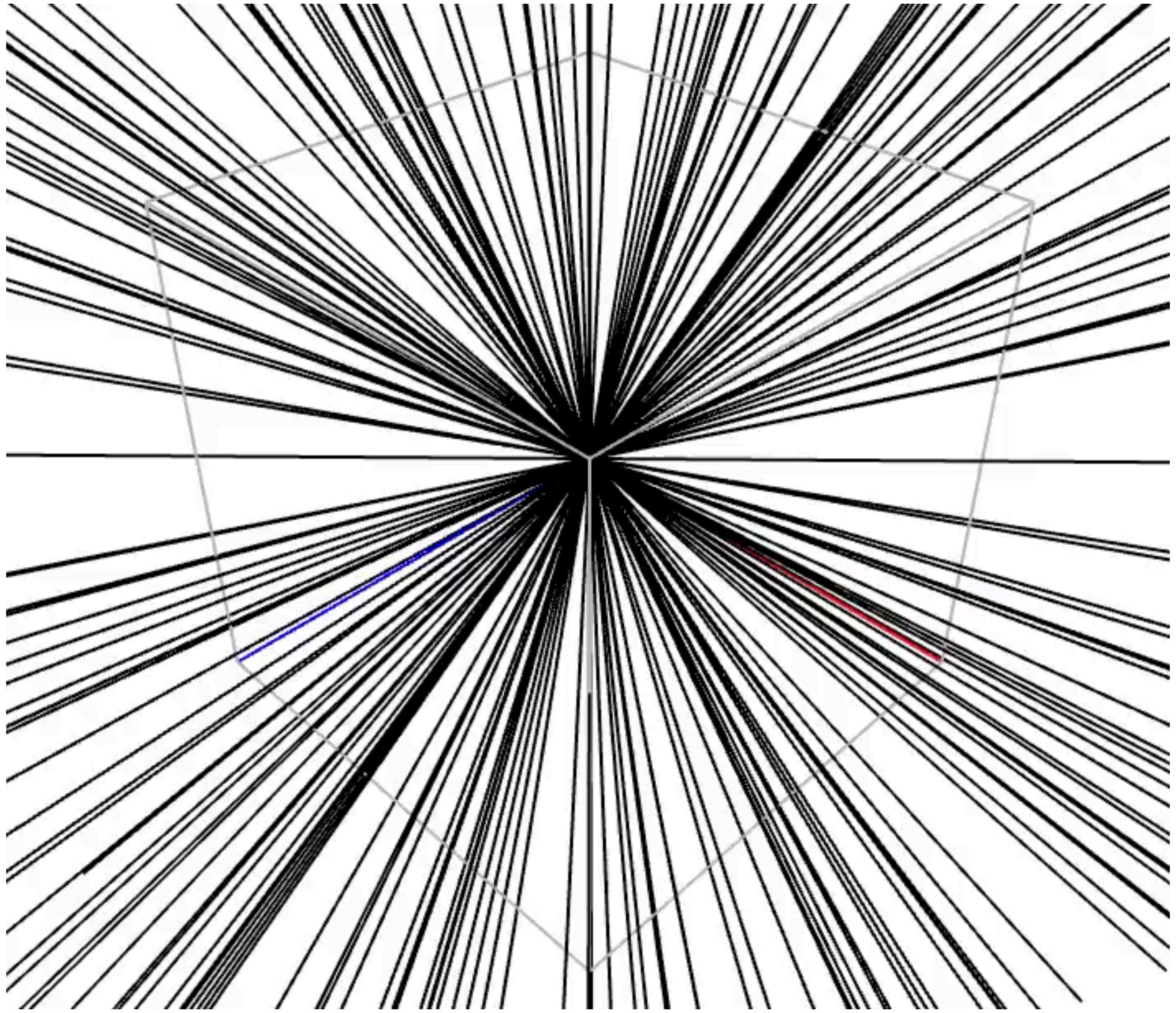
The Closest Flat Problem is Hard

- It’s not convex. How hard is it?
- Generate random 3D lines that intersect a known line. Can we recover the known line from a random initial guess?



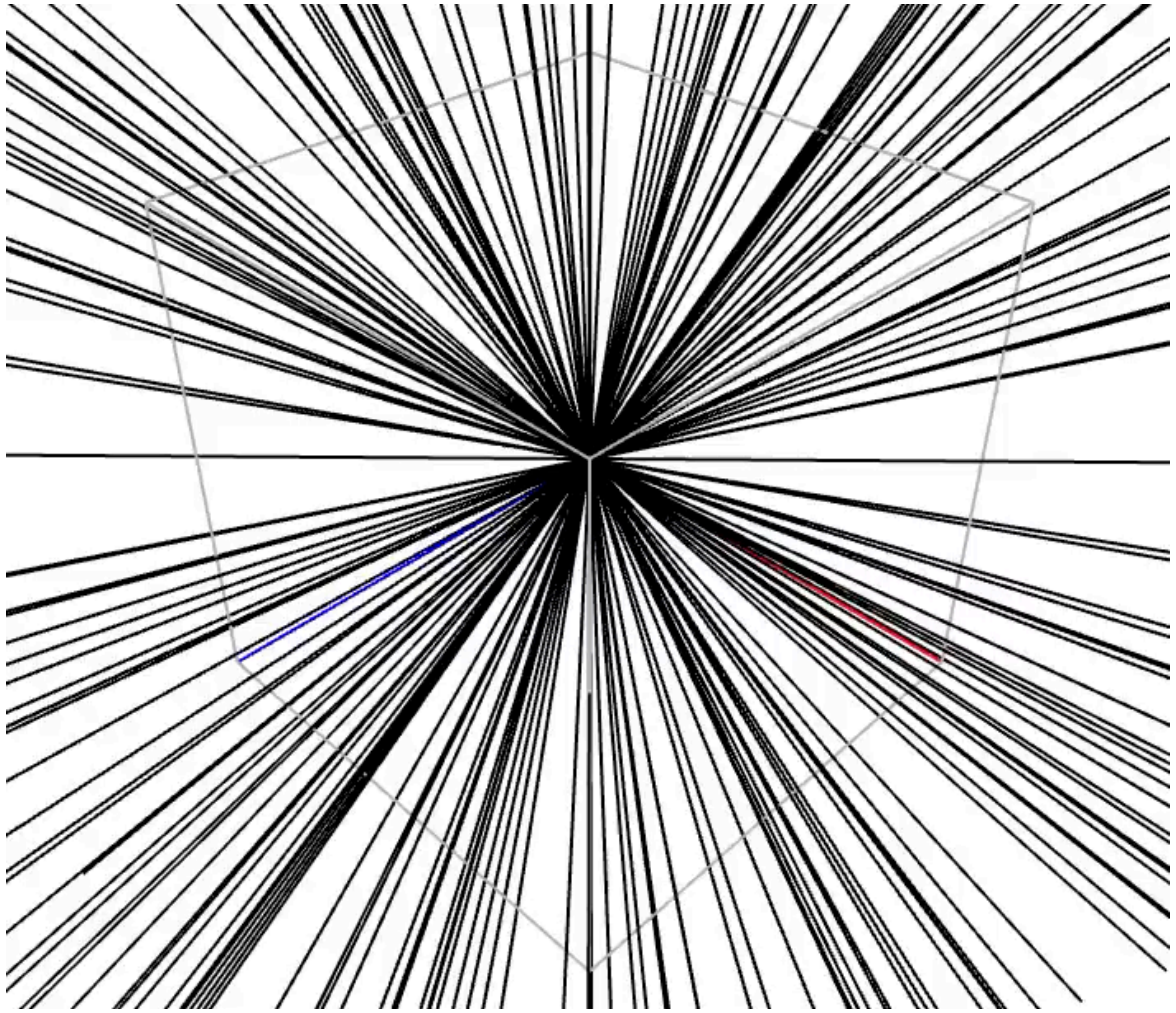
The Closest Flat Problem is Hard

- It’s not convex. How hard is it?
- Generate random 3D lines that intersect a known line. Can we recover the known line from a random initial guess?
- In 3D, the closest line to a set of lines.



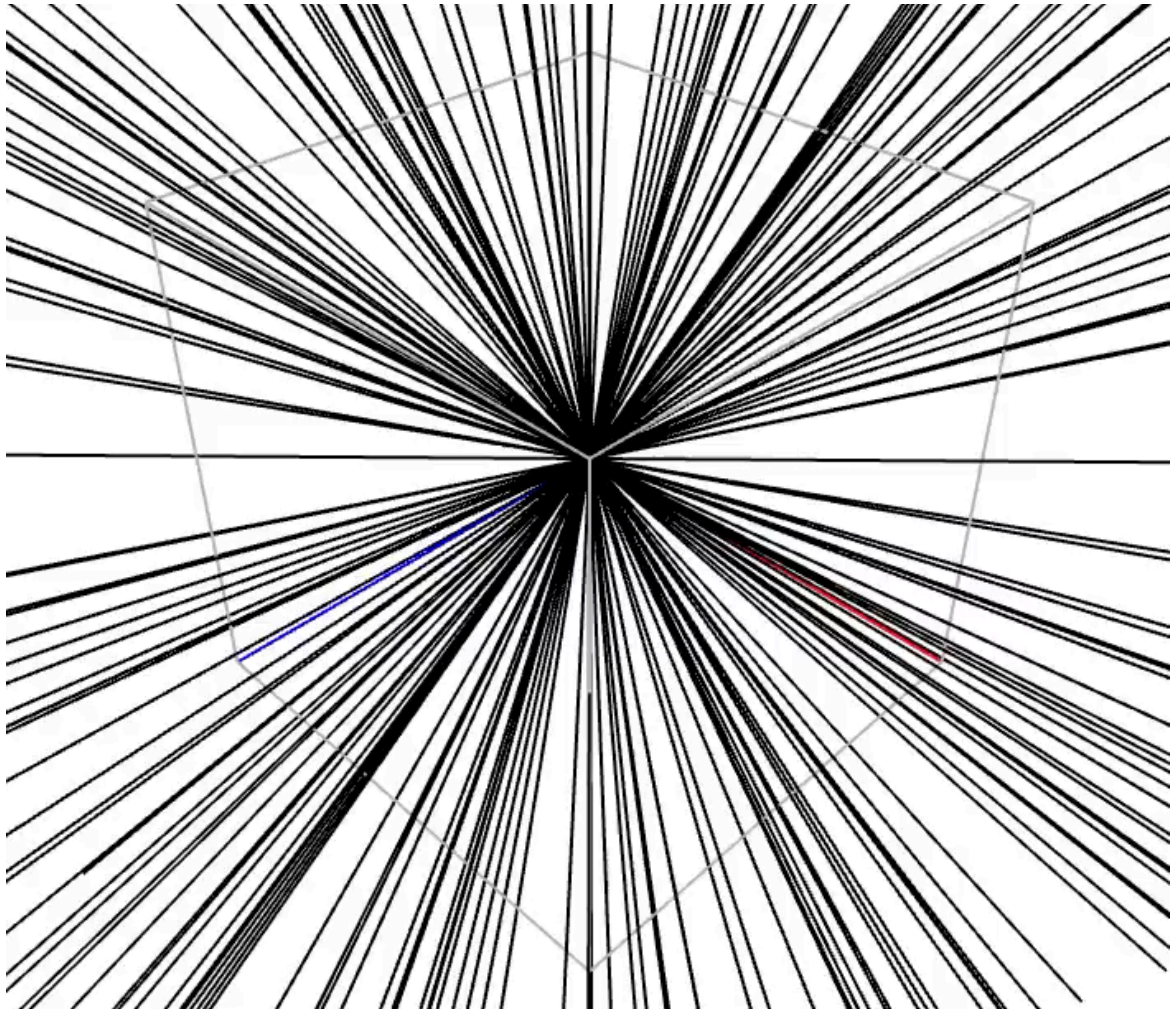
The Closest Flat Problem is Hard

- It's not convex. How hard is it?
- Generate random 3D lines that intersect a known line. Can we recover the known line from a random initial guess?
- In 3D, the closest line to a set of lines.
  - **Closest line** optimization as seen from a camera looking along the ground truth line: (the ground truth line looks like a point at the origin)



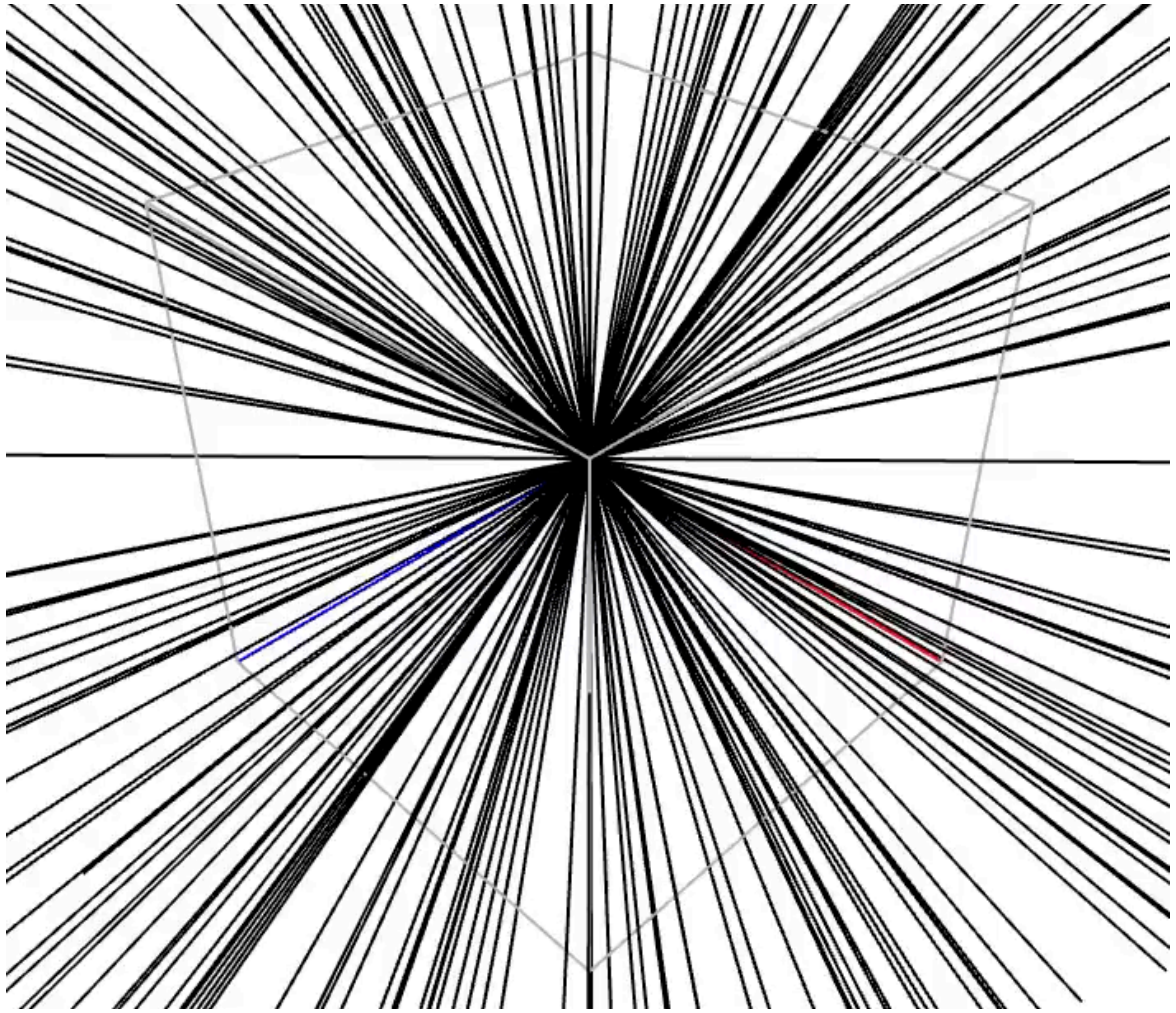
The Closest Flat Problem is Hard

- It's not convex. How hard is it?
- Generate random 3D lines that intersect a known line. Can we recover the known line from a random initial guess?
- In 3D, the closest line to a set of lines.
  - Closest line optimization as seen from a camera looking along the ground truth line: (the ground truth line looks like a point at the origin)



The Closest Flat Problem is Hard

- It's not convex. How hard is it?
- Generate random 3D lines that intersect a known line. Can we recover the known line from a random initial guess?
- In 3D, the closest line to a set of lines.
  - Closest line optimization as seen from a camera looking along the ground truth line: (the ground truth line looks like a point at the origin)
  - **Success!**



The Closest Flat Problem is Hard

The Closest Flat Problem is Hard

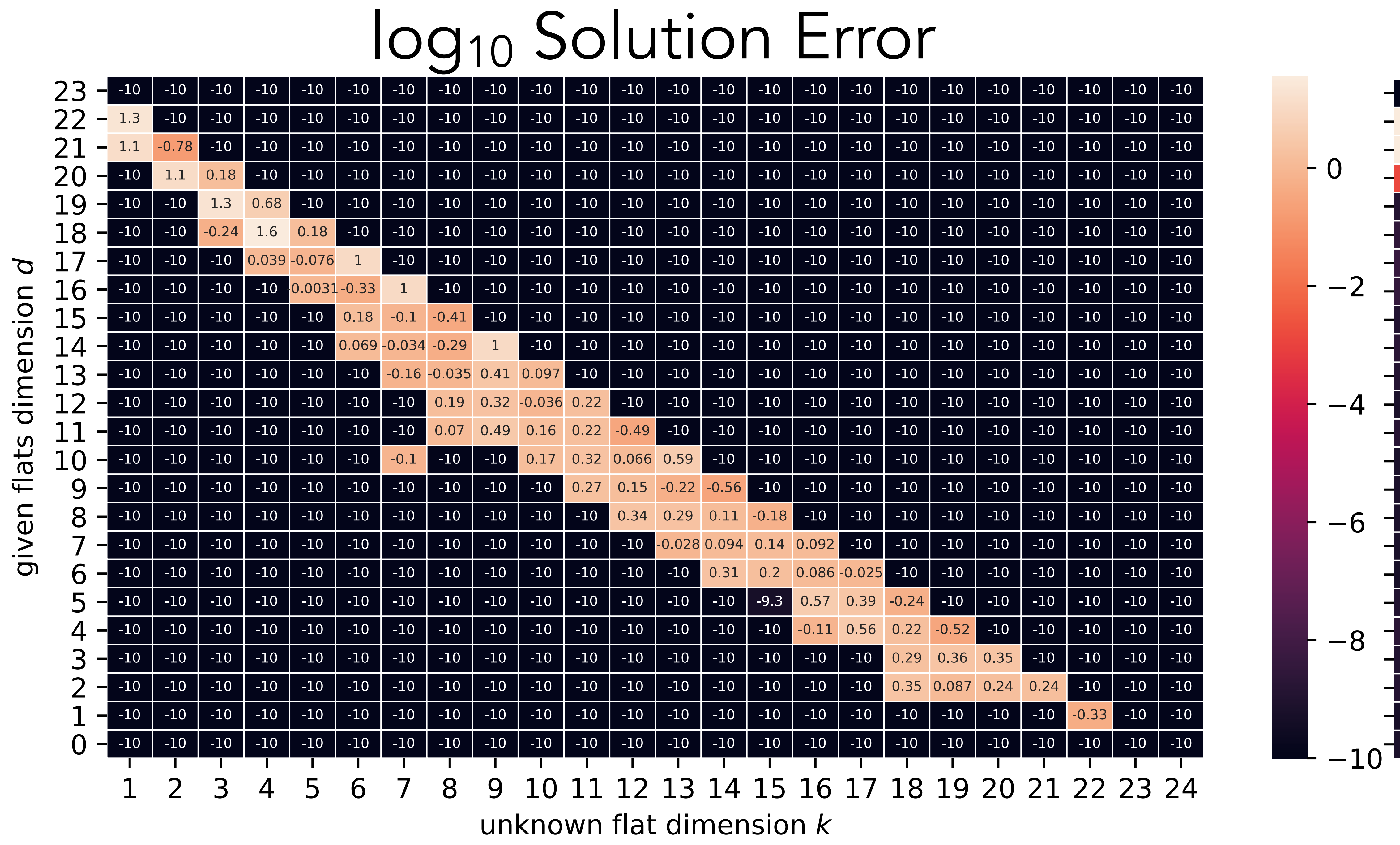
- An experiment in $\mathbb{R}^{24}$

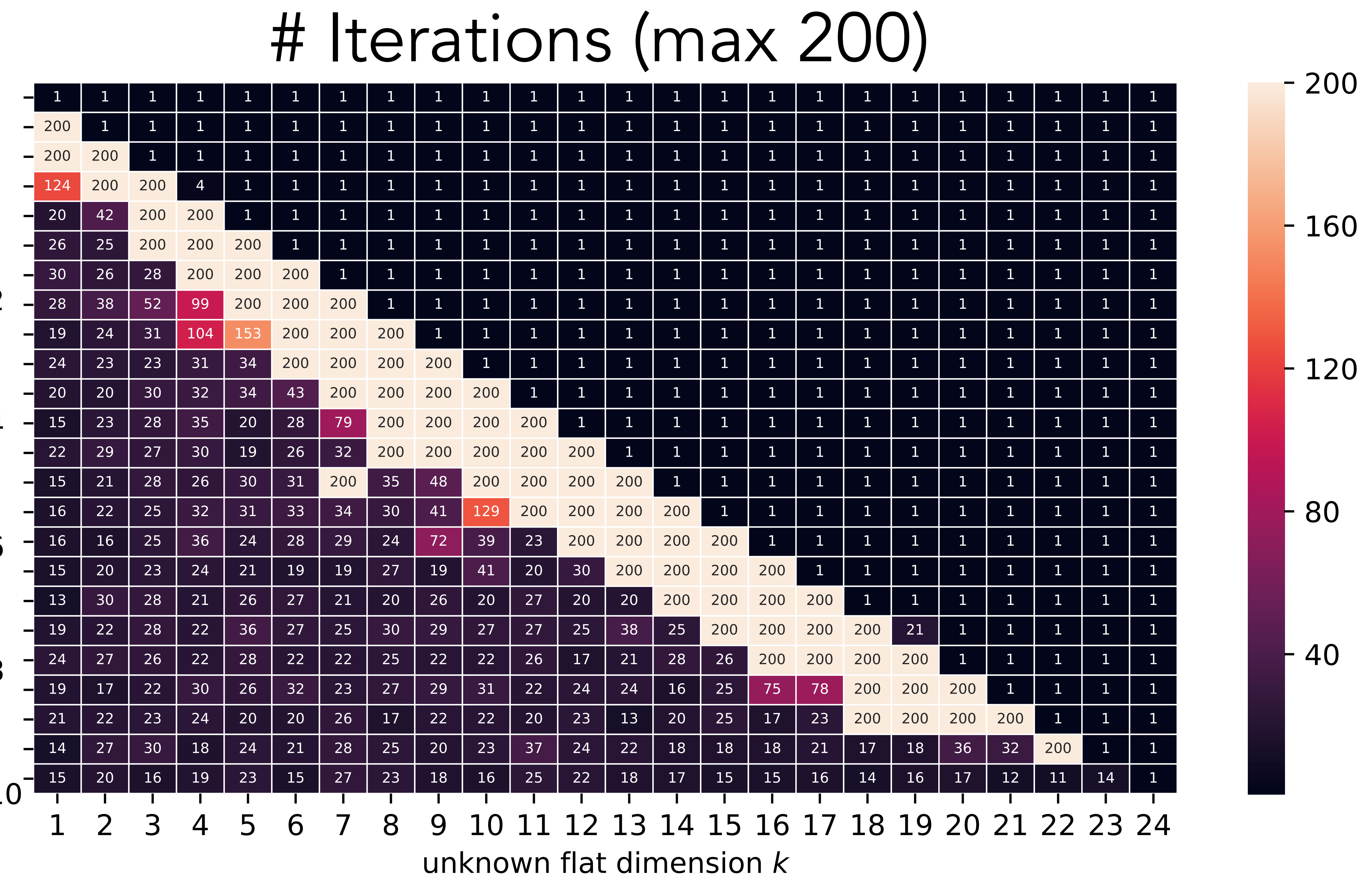
The Closest Flat Problem is Hard

- An experiment in $\mathbb{R}^{24}$
- Generate random *d*-dimensional flats that intersect a known *k*-dimensional flat. Can we recover the *k*-dimensional flat from a random initial guess?

The Closest Flat Problem is Hard

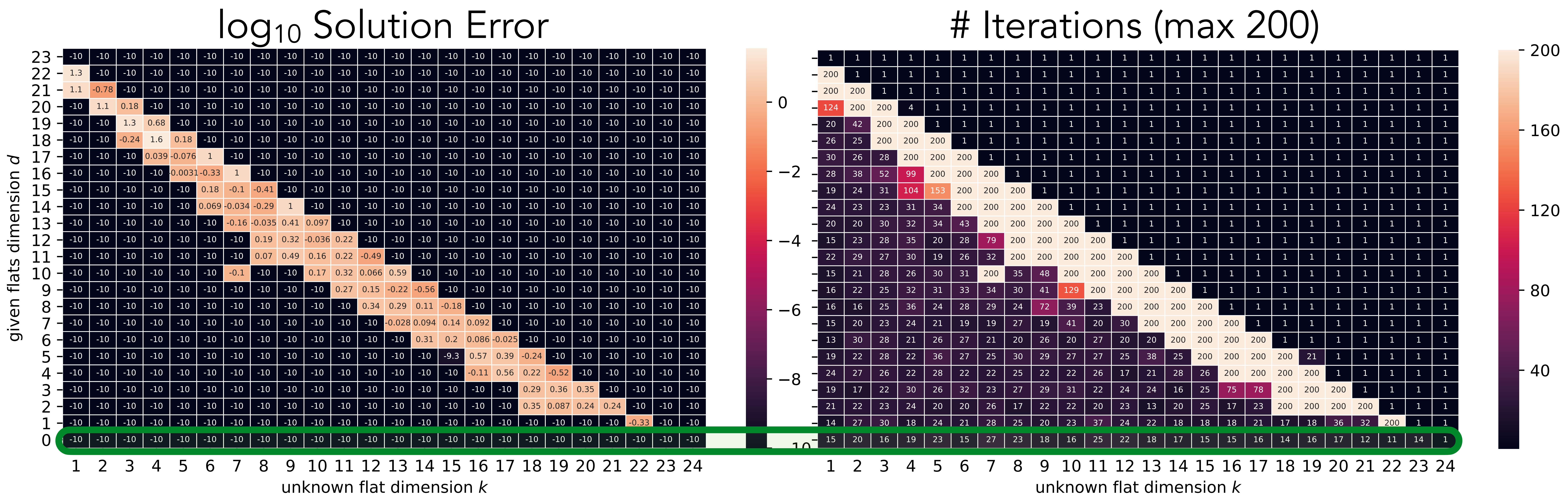
- An experiment in $\mathbb{R}^{24}$
- Generate random *d*-dimensional flats that intersect a known *k*-dimensional flat. Can we recover the *k*-dimensional flat from a random initial guess?





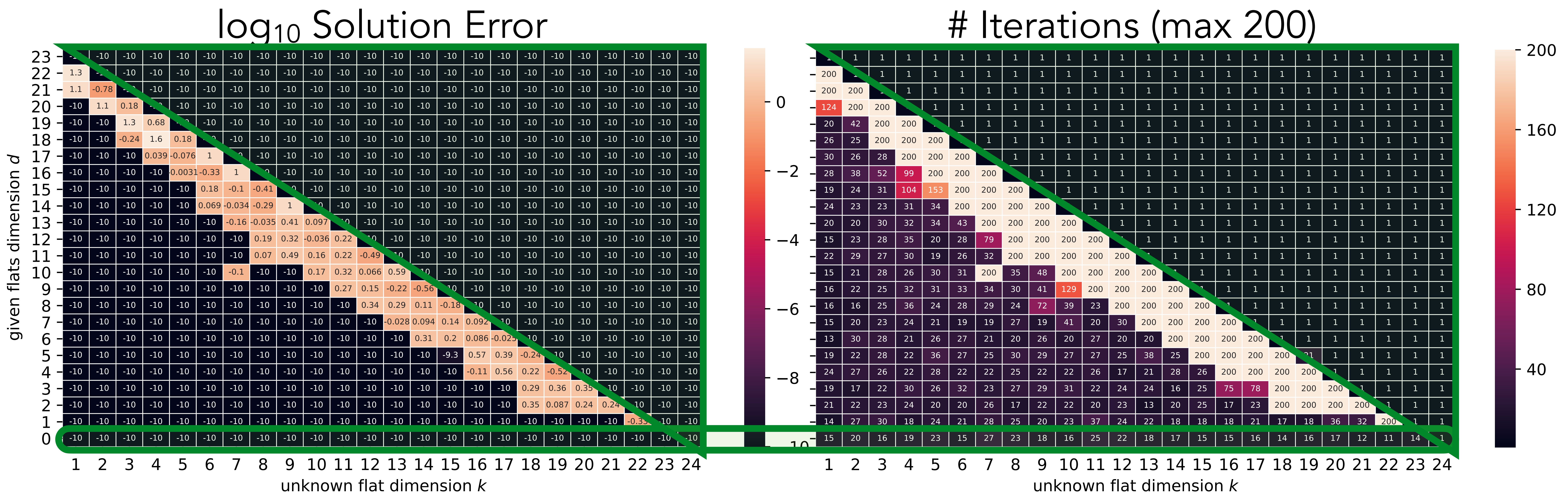
The Closest Flat Problem is Hard

- An experiment in $\mathbb{R}^{24}$
- Generate random d-dimensional flats that intersect a known k-dimensional flat. Can we recover the k-dimensional flat from a random initial guess?
  - When d=0, the given flats are points. It's a simple least squares problem



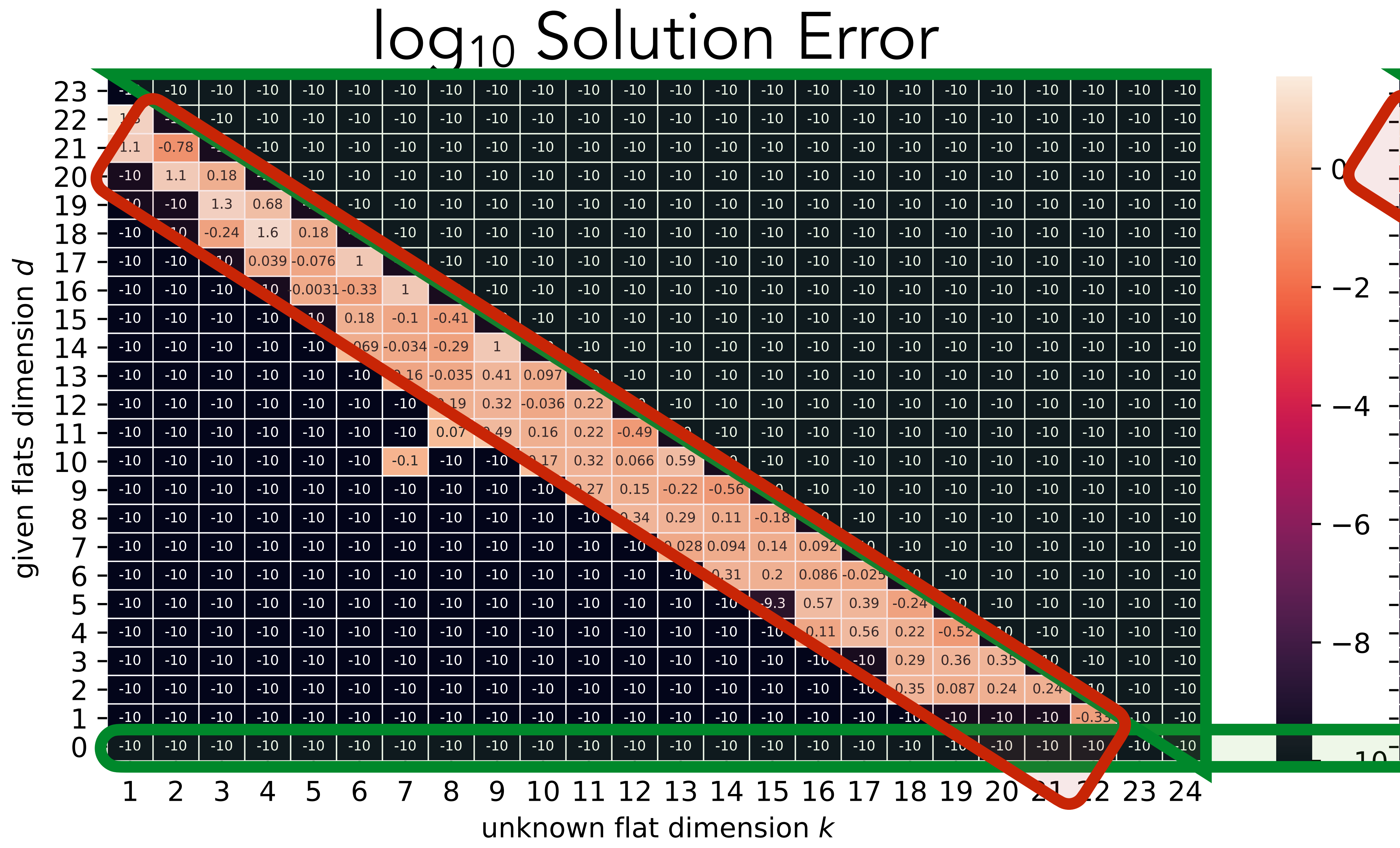
The Closest Flat Problem is Hard

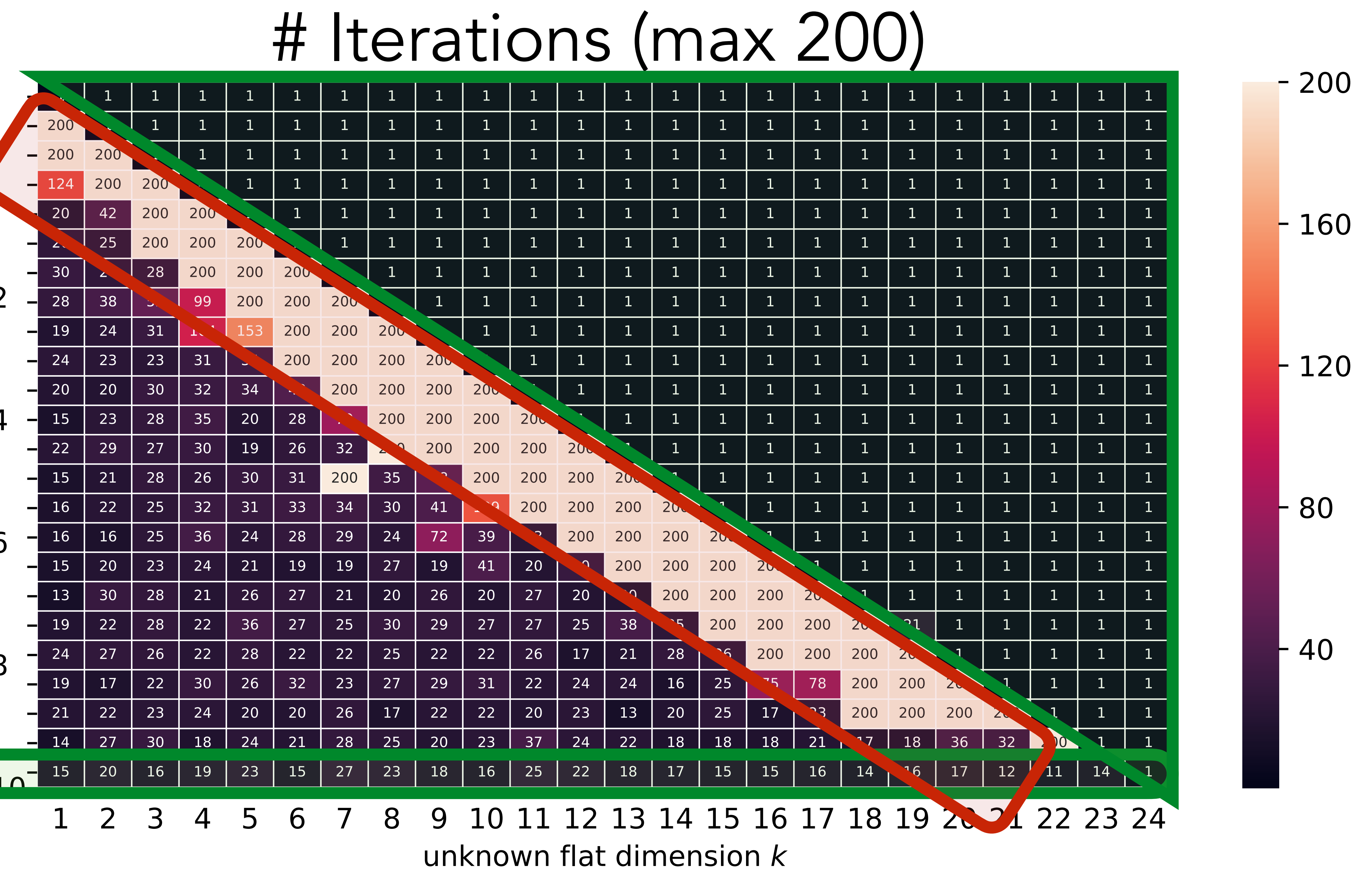
- An experiment in $\mathbb{R}^{24}$
- Generate random *d*-dimensional flats that intersect a known *k*-dimensional flat. Can we recover the *k*-dimensional flat from a random initial guess?
  - When d=0, the given flats are points. It's a simple least squares problem
  - When $d+k \geq 24$, it's trivial. A random initial guess almost surely intersects all flats.



The Closest Flat Problem is Hard

- An experiment in $\mathbb{R}^{24}$
- Generate random *d*-dimensional flats that intersect a known *k*-dimensional flat. Can we recover the *k*-dimensional flat from a random initial guess?
  - When $d=0$, the given flats are points. It's a simple least squares problem
  - When $d+k \geq 24$, it's trivial. A random initial guess almost surely intersects all flats.
  - When $d+k<24$, there is a difficult zone as $d+k$ approach 24.





The Closest Flat Problem is Hard

The Closest Flat Problem is Hard

- Tried many possibilities
  - direct gradient and Hessian-based optimization for an explicit representation of the flat
  - optimization on the Graff manifold
  - gradient-based optimization of projection matrices
  - global optimization via basin hopping
  - Karcher mean
  - alternating optimization strategies

The Closest Flat Problem is Hard

- Tried many possibilities
  - direct gradient and Hessian-based optimization for an explicit representation of the flat
  - optimization on the Graff manifold
  - gradient-based optimization of projection matrices
  - global optimization via basin hopping
  - Karcher mean
  - alternating optimization strategies
- See our Appendix "How Not to Minimize Flat/Flat Distances"

Table 7: The error resulting from various initial guess schemes followed by 10 iterations of our bi-quadratic flat optimization (Section 4.2) compared with ground truth. In this experiment, we keep the 50% of per-vertex initial guesses with lowest position error.

Model	Transformation Errors / Vertex Errors E_{RMS}						without initial guess
	Unconstrained approaches			Constrained approaches			
	One-ring	Euclidian	Geodesic	One-ring	Euclidian	Geodesic	
cylinder	0.01 / 0.88	0.21 / 13.52	0.52 / 18.42	0.01 / 1.48	0.2 / 12.91	0.58 / 20.1	0.45 / 31.29
cube	0.10 / 6.12	0.11 / 6.59	0.28 / 10.76	0.11 / 5.97	0.09 / 7.96	0.34 / 9.52	0.2 / 15.51
cheburashka	0.02 / 0.92	0.04 / 0.83	1.02 / 1.59	0.02 / 0.83	0.03 / 0.91	0.99 / 2.15	0.1 / 1.22
wolf	0.2 / 1e-8	1.12 / 6.7e-8	2.19 / 1.3e-4	0.2 / 6.7e-9	2.11 / 1.1e-6	0.58 / 2.7e-5	0.32 / 5e-10
cow	0.42 / 0.2	5.53 / 0.27	10.92 / 0.98	0.27 / 0.22	1.46 / 0.31	18.57 / 1.76	1.63 / 0.74

tion 4.2). In this experiment, we perform PCA on the the 50% of per-vertex initial guesses with lowest position error. The unconstrained one-ring neighborhood outperformed the other strategies. As a result of our experiments, and owing to its simplicity and run-time performance, we use the one-ring neighborhood with unconstrained 3D error (8) for our results.

Appendix C: How Not to Minimize Flat/Flat Distances

We seek to minimize the sum of squared flat/flat distances (Eq. 6) given an initial guess $\mathcal{L}_{guess}$. This minimization can be expressed in numerous ways. See Figure 5 for a comparison where relevant.

Direct optimization $(\mathbf{p}, B)$ We directly optimize Equation 6 using the BFGS algorithm [NW06]. This never achieves the low error of our proposed bi-quadratic approach. We also experimented with a combination of these two approaches, where we improve the bi-quadratic solution with direct optimization or switch approaches every 10 iterations. These combinations were inferior to simply running the bi-quadratic approach for additional iterations.

Optimization on an appropriate manifold ($\mathbf{p}, B$ manifold) We optimize Equation 6 with a various algorithms (gradient descent, conjugate gradient, and trust region) on the space of $\mathbb{R}^n \times \mathbf{Gr}(h-1, \mathbb{R}^n)$ [EAS98, TKW16]. The gradient descent and conjugate gradient algorithms are slower to compute and achieve higher error per iteration than our proposed bi-quadratic approach. The Hessian-based trust region algorithm is much slower to compute, taking hours to execute 20 iterations. However, on our simplest example, a cylinder with four bones, the trust region algorithm achieves superlinear convergence and the known ground truth solution (Figure 5).

Global optimization We employed basin hopping [WD97], which is a stochastic global minimization algorithm in which random modifications of the current state are optimized via continuous optimization. We used our proposed approach (Section 4.2) for the continuous optimization. Basin hopping failed to improve upon the error of our proposed approach alone. The random modifications did not find basins with lower error. This approach is not plotted in Figure 5, because the curve would cover that of our proposed bi-quadratic approach.

Karcher Mean We experimented with computing the Riemannian center of mass or Karcher mean of the given flats. The Karcher mean was proposed in the literature [CHV17, MRBD*14] as an effective technique for finding the centroid to a set of points on a Riemannian manifold. We experimented with representing flats as points on (a) the product manifold $\mathbb{R}^n \times \mathbf{Gr}(h-1, \mathbb{R}^n)$ or (b) the Graff manifold identified with points on the higher-dimensional Grassmann manifold (Appendix A). In our setting, the unknown flat has different dimension than the given flats; in this case, the additional principal angles needed for the geodesic distance computation are taken as $\frac{\pi}{2}$. Unfortunately, this approach does not find a flat with small distance to other flats. We believe that this is due to the distortion of distances on the product or Graff manifolds.

Iterative PCA (IPCA) We optimize Equation 6 with a different alternating decomposition than our proposed bi-quadratic approach. Instead, we alternate between (a) solving for the closest point on each vertex's flat to the handle flat $\mathcal{L}$ and then (b) solving for the flat that minimizes the squared distance to these closest points. Step (a) can be solved via

$$\underset{\mathbf{x}}{\operatorname{argmin}}\ (\mathbf{x}-\mathbf{p})^\top P_{null}(\mathbf{x}-\mathbf{p}) \tag{18}$$

subject to:

$$\bar{V}_i \mathbf{x} = \mathbf{v}_i' \tag{19}$$

where $P_{null} = I_{3\cdot\#\text{poses}} - B(B^\top B)^{-1}B^\top = I_{3\cdot\#\text{poses}} - BB^\dagger$ is the orthogonal projector onto the null-space of the handle flat (Appendix A) and $B^\dagger$ is the Moore-Penrose pseudo-inverse of B. This requires solving a different $(3\cdot\#\text{poses}) \times (3\cdot\#\text{poses})$ system of equations for each vertex, with the constraint implemented either via Lagrange multipliers or as a least squares soft constraint. Step (b) can be solved by principal component analysis (PCA), taking the first $h-1$ principal components as the parallel directions for the handle flat and the center as the point through which the handle flat passes.

This iterative PCA (IPCA) approach produces better results than all other techniques except for our bi-quadratic approach (and the very expensive Hessian-based trust region approach). Our bi-quadratic approach alternates between (a) solving for the closest point on the handle flat $\mathcal{L}$ to each vertex's flat (in terms of handle flat parameters $\mathbf{w}_i$) and (b) solving for a new handle basis matrix F that minimizes the distance to the vertex flats using the $\mathbf{w}_i$ parameters. Our bi-quadratic approach is faster to compute, as it only requires the solution to a single, smaller $4h \times 4h$ system of equations.

Iterative Laplacian re-weighting Any point on a d-dimensional flat can be represented as the weighted average of $d+1$ or more affine independent points. In our setting, this implies that the following energy for per-vertex transformation matrices $\mathbf{t}_i \in \mathbb{R}^{12\cdot\#\text{poses}}$ should be zero:

$$E_{\text{local}} = \sum_i \|\mathbf{t}_i - \sum_{j\in\mathcal{N}(i)} w_{ij}\mathbf{t}_j\|^2 \tag{20}$$

where $\mathcal{N}(i)$ are the neighbors of vertex i and w_{ij} are scalar weights that sum to one. E_{local} can be expressed as $E_{\text{local}} = \sum_i \|L\bar{\mathbf{t}}\|^2$, where L is a $12\cdot\#\text{pose}\cdot\#\text{vertices}$ laplacian matrix and $\bar{\mathbf{t}}$ is a column vector containing all vertices' transformation matrices across all poses. We experimented with two definitions of vertex neighbors: the one-ring; and a fixed, random set of $2h$ vertices. To reproduce the observed poses, we wish to minimize:

$$E_{\text{data}} = \sum_i \|\bar{V}_i\mathbf{t}_i - \mathbf{v}_i'\|^2 \tag{21}$$

S. Liu & J. Tan & Z. Deng & Y. Gingold

We optimize the sum of the two terms. The expression $E_{\text{local}} + E_{\text{data}}$ is quadratic in either w_{ij} or $\mathbf{t}_i$, so we alternate between solving for one while fixing the other. When solving for w_{ij}, E_{data} is constant and can be ignored, resulting in a small, typically underdetermined, local system per-vertex that can be solved in a least-square sense. Solving for $\mathbf{t}_i$, however, amounts to solving a very large, sparse system of equations. Finally, we take the first $h-1$ principal components of the final $\mathbf{t}_i$ to be the handle flat.

Because of the very large system of equations, this approach executes much more slowly than our proposed bi-quadratic approach and produces solutions with more error per iteration.

Orthogonal Projector The minimal distance between flats (Equation 5) can be written $\|C(x_0 - y_0)\|$, where x_0 is any point on one flat and y_0 is any point on the other flat and C is the projection matrix onto the intersection of the two flats' orthogonal spaces [DK92]. For our problem, this results in the expression:

$$\sum_i \|C_i(\mathbf{p}-\mathbf{t}_i)\|^2 = \mathbf{p}^\top\left(\sum_i C_i\right)\mathbf{p} + \left(\sum_i \mathbf{t}_i^\top C_i\mathbf{t}_i\right) - 2\mathbf{p}^\top\left(\sum_i C_i\mathbf{t}_i\right) \tag{22}$$

where the $\mathbf{t}_i$ are any valid transformation matrix in vertex i's flat (Equation 7). The projection matrix C_i can be written (via the Anderson-Duffin formula) as $C_i = 2P_{\bar{V}_i}(P_{\bar{V}_i} + P_B)^\dagger P_B$, where P_B and $P_{\bar{V}_i}$ are orthogonal projectors onto the column-space of B and the row-space of $\bar{V}_i$, respectively. This approach is unstable and tends to increase error from a good initial guess.

Equation 22 is minimized (by setting the derivative with respect to $\mathbf{p}$ to 0) when $\mathbf{p} = (\sum_i C_i)^{-1}(\sum_i C_i\mathbf{t}_i)$. Substituting this expression for $\mathbf{p}$ results in:

$$\min_B \left(\sum_i \mathbf{t}_i^\top C_i\mathbf{t}_i\right) - \left(\sum_i C_i\mathbf{t}_i\right)^\top\left(\sum_i C_i\right)^{-1}\left(\sum_i C_i\mathbf{t}_i\right) \tag{23}$$

This expression is numerically unstable, because C_i is rank deficient. This rank deficiency corresponds to the fact that p can be any point on a flat. Even with a pseudoinverse, the expression is unstable.

Minimizing Flat/Flat Distance: Initial Guess

Minimizing Flat/Flat Distance: Initial Guess

- Find an $\mathbb{R}^{12p}$ point $\mathbf{x}_i$ for each vertex

Minimizing Flat/Flat Distance: Initial Guess

- Find an $\mathbb{R}^{12p}$ point $\mathbf{x}_i$ for each vertex
    - If the vertex $\mathbf{v}_i$ and its one-ring move rigidly, there is a unique solution. If not, there is a least-squares solution…

Minimizing Flat/Flat Distance: Initial Guess

- Find an $\mathbb{R}^{12p}$ point $\mathbf{x}_i$ for each vertex
  - If the vertex $\mathbf{v}_i$ and its one-ring move rigidly, there is a unique solution. If not, there is a least-squares solution…
    - …measuring error in $\mathbb{R}^{12p}$:

$$\mathbf{x}_i = \operatorname*{argmin}_{\mathbf{x}} \sum_{j \in \{i\} \cup \mathcal{N}(i)} \left\| \frac{1}{\|\mathbf{v}_j\|^2} \bar{V}_j^\top \bar{V}_j (\mathbf{x} - \mathbf{t}_j) \right\|^2$$

Minimizing Flat/Flat Distance: Initial Guess

- Find an $\mathbb{R}^{12p}$ point $\mathbf{x}_i$ for each vertex
  - If the vertex $\mathbf{v}_i$ and its one-ring move rigidly, there is a unique solution. If not, there is a least-squares solution…
    - …measuring error in $\mathbb{R}^{12p}$:

$$\mathbf{x}_i = \operatorname*{argmin}_{\mathbf{x}} \sum_{j \in \{i\} \cup \mathcal{N}(i)} \left\| \frac{1}{\|\mathbf{v}_j\|^2} \bar{V}_j^\top \bar{V}_j (\mathbf{x} - \mathbf{t}_j) \right\|^2$$

    - …measuring error in 3D:

$$\mathbf{x}_i = \operatorname*{argmin}_{\mathbf{x}} \sum_{j \in \{i\} \cup \mathcal{N}(i)} \|\bar{V}_j \mathbf{x} - \mathbf{v}'_j\|^2$$

Minimizing Flat/Flat Distance: Initial Guess

- Find an $\mathbb{R}^{12p}$ point $\mathbf{x}_i$ for each vertex
  - If the vertex $\mathbf{v}_i$ and its one-ring move rigidly, there is a unique solution. If not, there is a least-squares solution…
    - …measuring error in $\mathbb{R}^{12p}$:

$$\mathbf{x}_i = \operatorname*{argmin}_{\mathbf{x}} \sum_{j \in \{i\} \cup \mathcal{N}(i)} \left\| \frac{1}{\|\mathbf{v}_j\|^2} \bar{V}_j^\top \bar{V}_j (\mathbf{x} - \mathbf{t}_j) \right\|^2$$

    - …measuring error in 3D:

$$\mathbf{x}_i = \operatorname*{argmin}_{\mathbf{x}} \sum_{j \in \{i\} \cup \mathcal{N}(i)} \|\bar{V}_j \mathbf{x} - \mathbf{v}'_j\|^2$$

- PCA on the 12p-dimensional points gives us an initial guess for the flat.

Minimizing Flat/Flat Distance: Optimization

- We use an explicit expression for a flat: $$\min_F \sum_i \|\bar{V}_i F \mathbf{w}_i - \mathbf{v}_i'\|^2$$

subject to: $\sum \mathbf{w}_i = 1$

Minimizing Flat/Flat Distance: Optimization

- We use an explicit expression for a flat: $$\min_F \sum_i \|\bar{V}_i F \mathbf{w}_i - \mathbf{v}'_i\|^2$$

  subject to: $\sum \mathbf{w}_i = 1$

- Quadratic in F, $\mathbf{w_i}$, and even $\bar{V}_i$.

Minimizing Flat/Flat Distance: Optimization

- We use an explicit expression for a flat: $$\min_F \sum_i \|\bar{V}_i F \mathbf{w}_i - \mathbf{v}_i'\|^2$$

  subject to: $\sum \mathbf{w}_i = 1$

- Quadratic in F, $\mathbf{w_i}$, and even $\bar{V}_i$.
- Alternates between local and global steps:

Minimizing Flat/Flat Distance: Optimization

- We use an explicit expression for a flat: $$\min_F \sum_i \|\bar{V}_i F \mathbf{w}_i - \mathbf{v}'_i\|^2$$

  subject to: $\sum \mathbf{w}_i = 1$

- Quadratic in F, $\mathbf{w_i}$, and even $\bar{V}_i$.
- Alternates between local and global steps:
  - Local steps: $\mathbf{w}_i$ are independent

Minimizing Flat/Flat Distance: Optimization

- We use an explicit expression for a flat:

$$\min_{F} \sum_{i} \|\bar{V}_i F \mathbf{w}_i - \mathbf{v}'_i\|^2$$

subject to: $\sum \mathbf{w}_i = 1$

- Quadratic in F, $\mathbf{w_i}$, and even $\bar{V}_i$.
- Alternates between local and global steps:
  - Local steps: $\mathbf{w}_i$ are independent
  - Global step: minimizing F entails solving a linear matrix equation:

$$\sum_{i} \left(I_{3 \cdot \#\text{poses}} \otimes (\mathbf{v}_i \mathbf{v}_i^\top) \right) F \left(\mathbf{w}_i \mathbf{w}_i^\top \right) = - \sum_{i} \bar{V}_i \mathbf{v}_i'^\top \mathbf{w}_i$$

Minimizing Flat/Flat Distance: Optimization

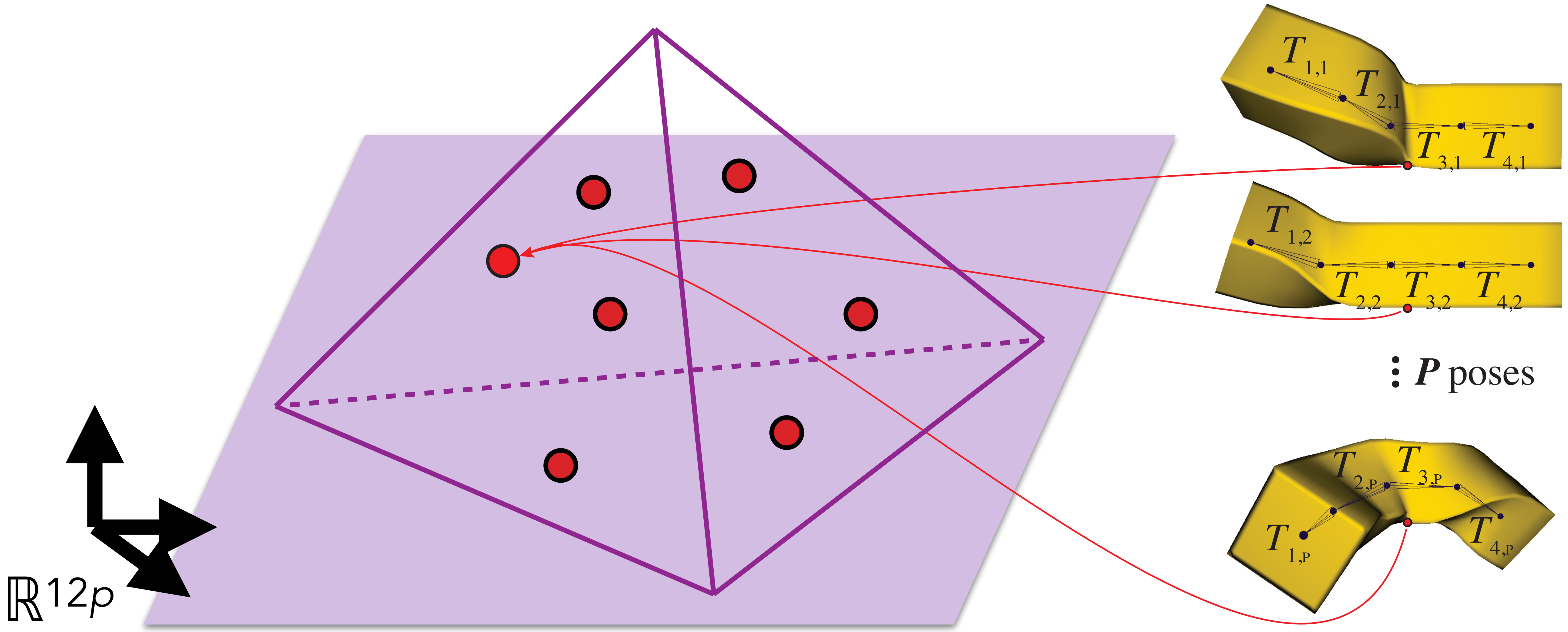
- We use an explicit expression for a flat: $$\min_F \sum_i \|\bar{V}_i F \mathbf{w}_i - \mathbf{v}'_i\|^2$$

  subject to: $\sum \mathbf{w}_i = 1$

- Quadratic in F, $\mathbf{w_i}$, and even $\bar{V}_i$.
- Alternates between local and global steps:
  - Local steps: $\mathbf{w}_i$ are independent
  - Global step: minimizing F entails solving a linear matrix equation: $$\sum_i \left(I_{3 \cdot \#\text{poses}} \otimes (\mathbf{v}_i \mathbf{v}_i^\top) \right) F \left(\mathbf{w}_i \mathbf{w}_i^\top \right) = - \sum_i \bar{V}_i \mathbf{v}_i'^\top \mathbf{w}_i$$
    - This reduces to a $4h \times 4h$ system of equations

Minimizing Flat/Flat Distance: Optimization

- Let's visualize optimization steps.
- A cylinder with 4 handles. The handle simplex is a tetrahedron. The handle flat is 3D. Let's visualize the closest points on the flat to the cylinder vertices.



Minimizing Flat/Flat Distance: Optimization

Minimizing Flat/Flat Distance: Optimization

- A cylinder with 4 handles

Minimizing Flat/Flat Distance: Optimization

- A cylinder with 4 handles

$\Rightarrow$ The handle simplex is a tetrahedron

Minimizing Flat/Flat Distance: Optimization

- A cylinder with 4 handles

  ⇒ The handle simplex is a tetrahedron

  ⇒ The handle flat is 3D

Minimizing Flat/Flat Distance: Optimization

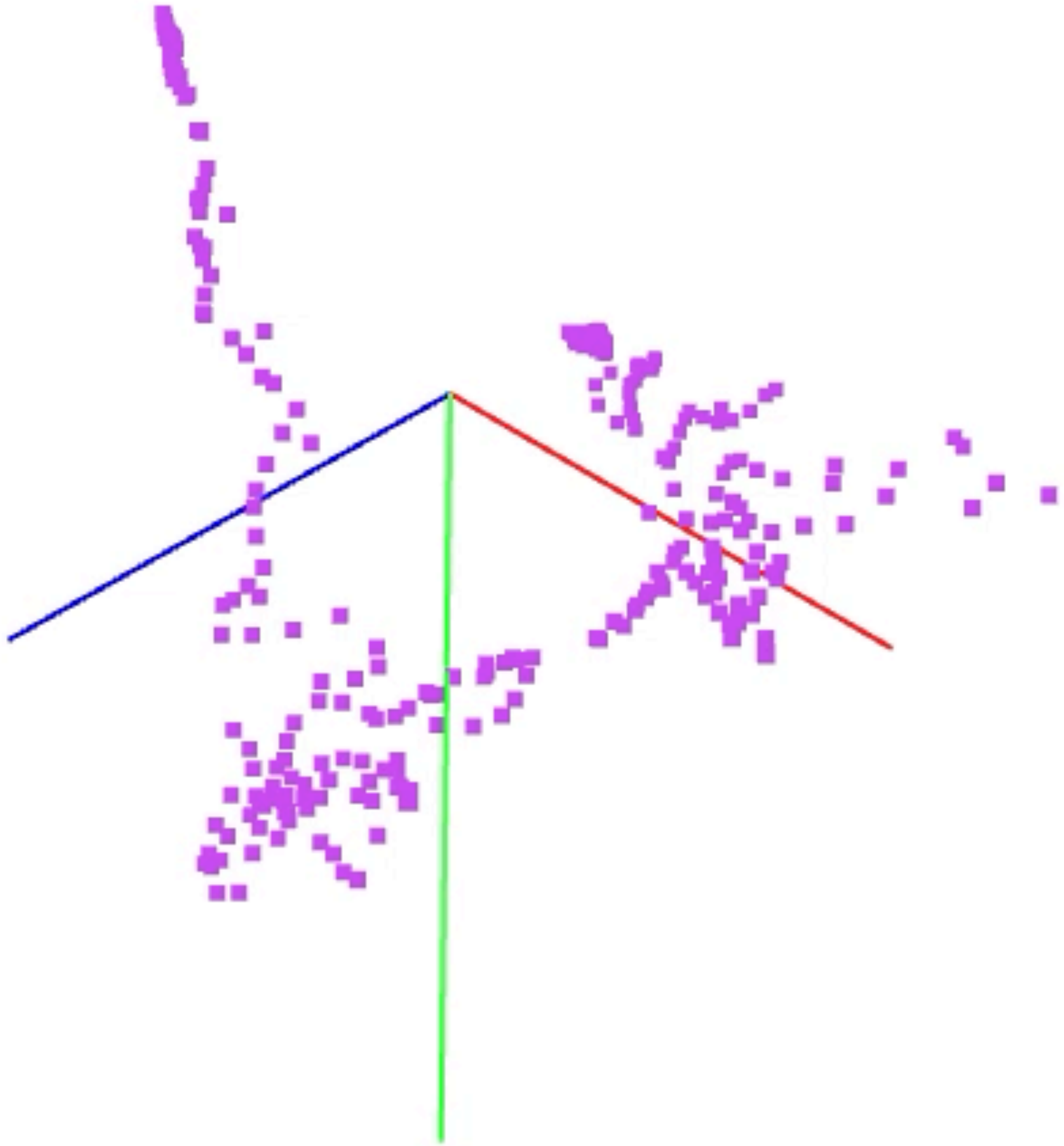
- A cylinder with 4 handles

  ⇒ The handle simplex is a tetrahedron

  ⇒ The handle flat is 3D
- Visualizing vertex transformations $\in \mathbb{R}^{12p}$ as points projected onto the handle flat:

Minimizing Flat/Flat Distance: Optimization

- A cylinder with 4 handles
  ⇒ The handle simplex is a tetrahedron
  ⇒ The handle flat is 3D
- Visualizing vertex transformations $\in \mathbb{R}^{12p}$ as points projected onto the handle flat:



Optimization from our initial guess

(slow motion, converges in ~10 iterations)

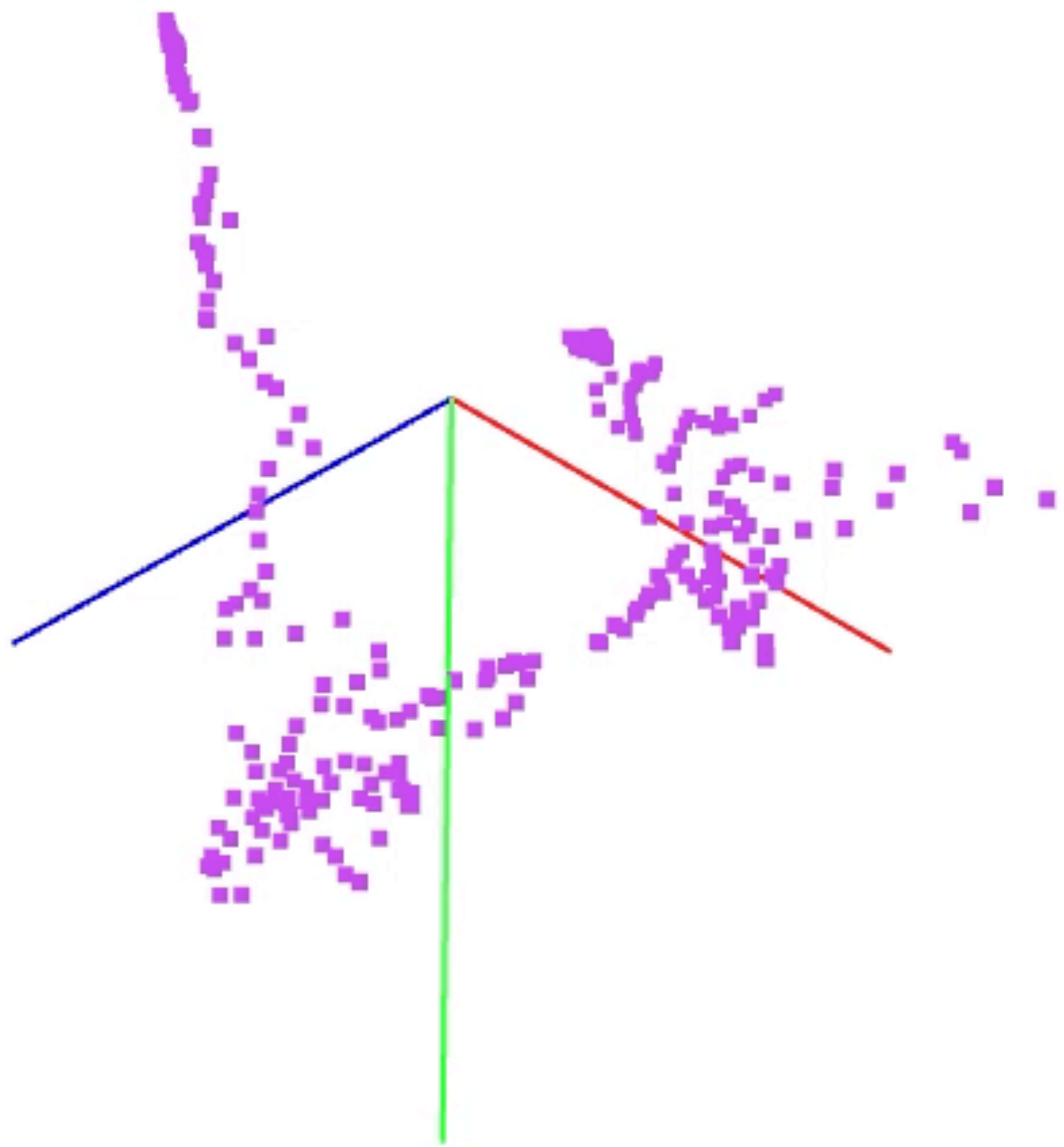
Minimizing Flat/Flat Distance: Optimization

- A cylinder with 4 handles

  ⇒ The handle simplex is a tetrahedron

  ⇒ The handle flat is 3D

- Visualizing vertex transformations $\in \mathbb{R}^{12p}$ as points projected onto the handle flat:

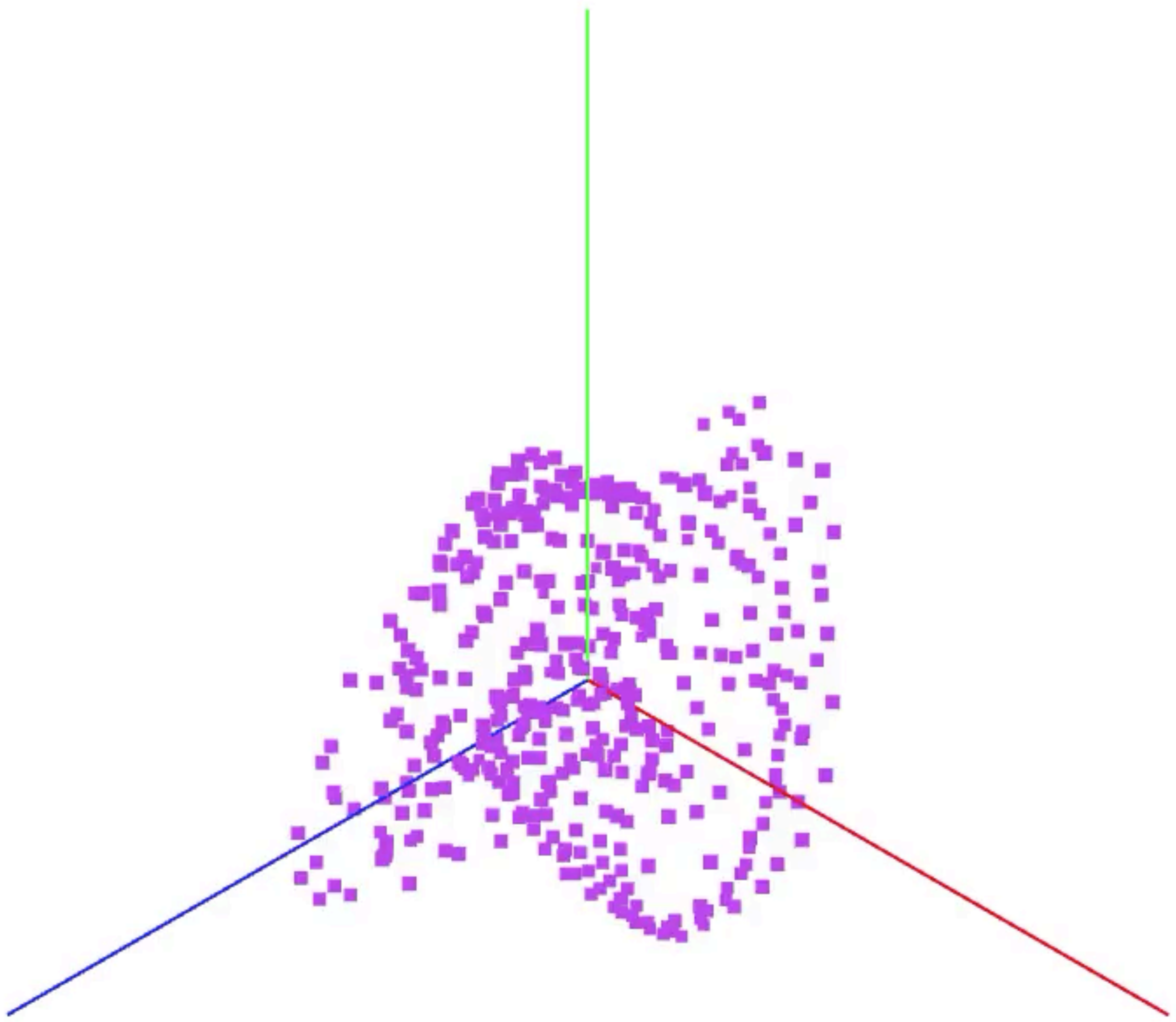


Optimization from our initial guess

(slow motion, converges in ~10 iterations)

Minimizing Flat/Flat Distance: Optimization

- A cylinder with 4 handles
  ⇒ The handle simplex is a tetrahedron
  ⇒ The handle flat is 3D
- Visualizing vertex transformations $\in \mathbb{R}^{12p}$ as points projected onto the handle flat:



Optimization from a random initial guess

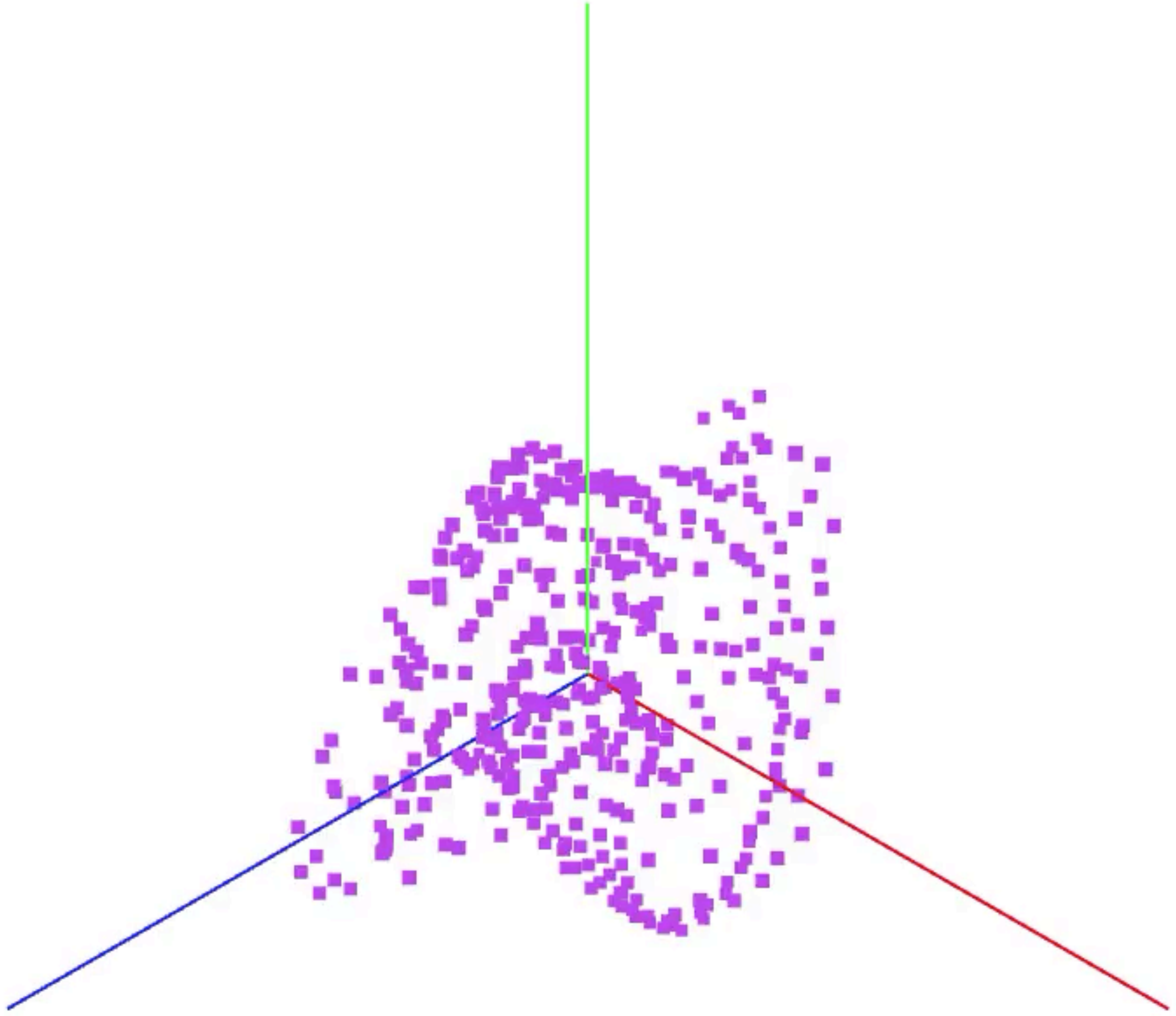
(>10,000 iterations)

Minimizing Flat/Flat Distance: Optimization

- A cylinder with 4 handles

  ⇒ The handle simplex is a tetrahedron

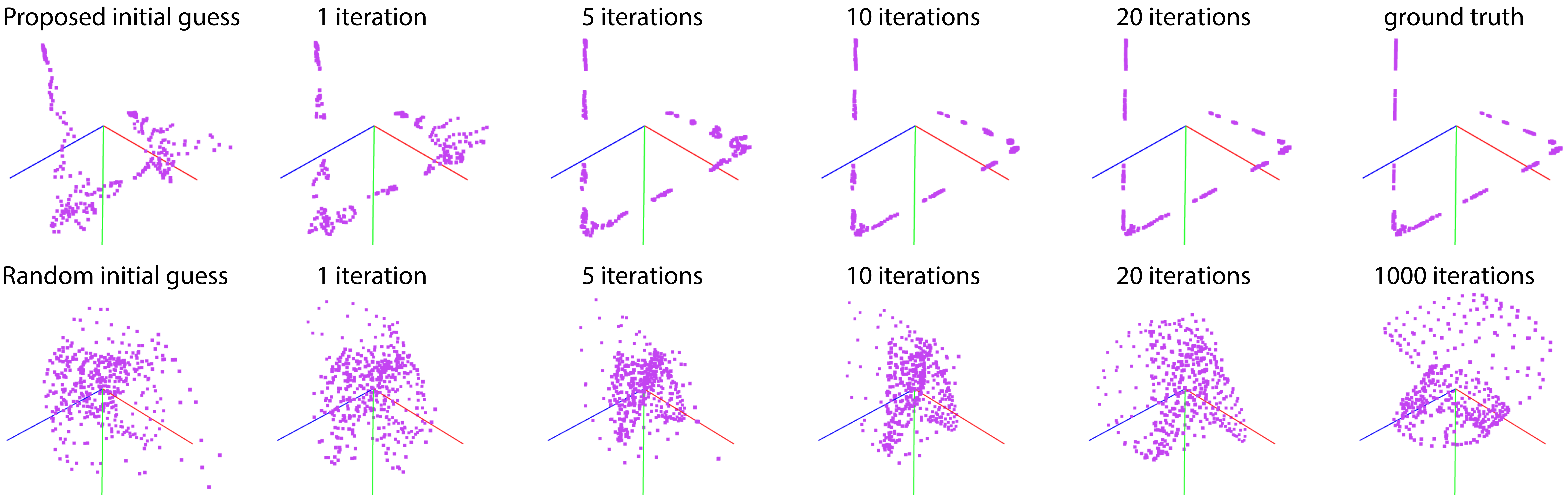
  ⇒ The handle flat is 3D
- Visualizing vertex transformations $\in \mathbb{R}^{12p}$ as points projected onto the handle flat:



Optimization from a random initial guess

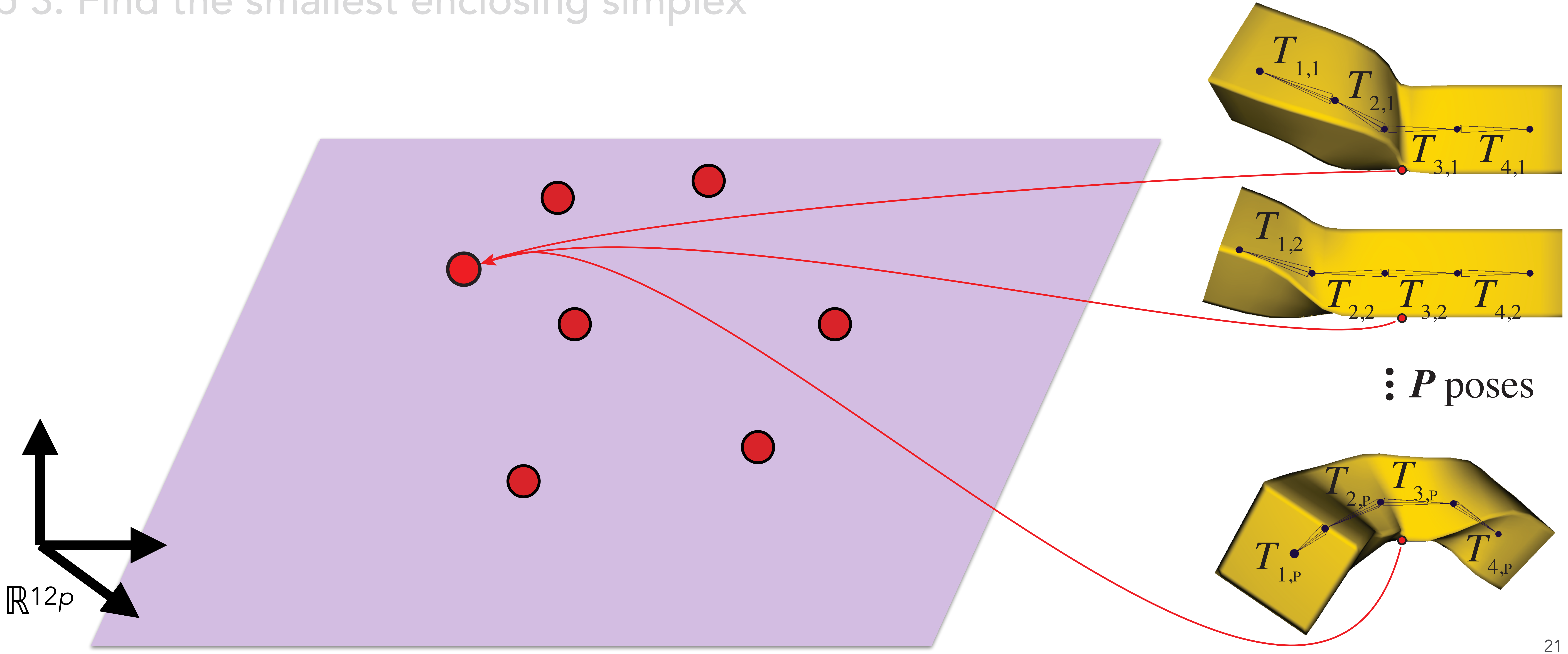
(>10,000 iterations)

Minimizing Flat/Flat Distance: Optimization



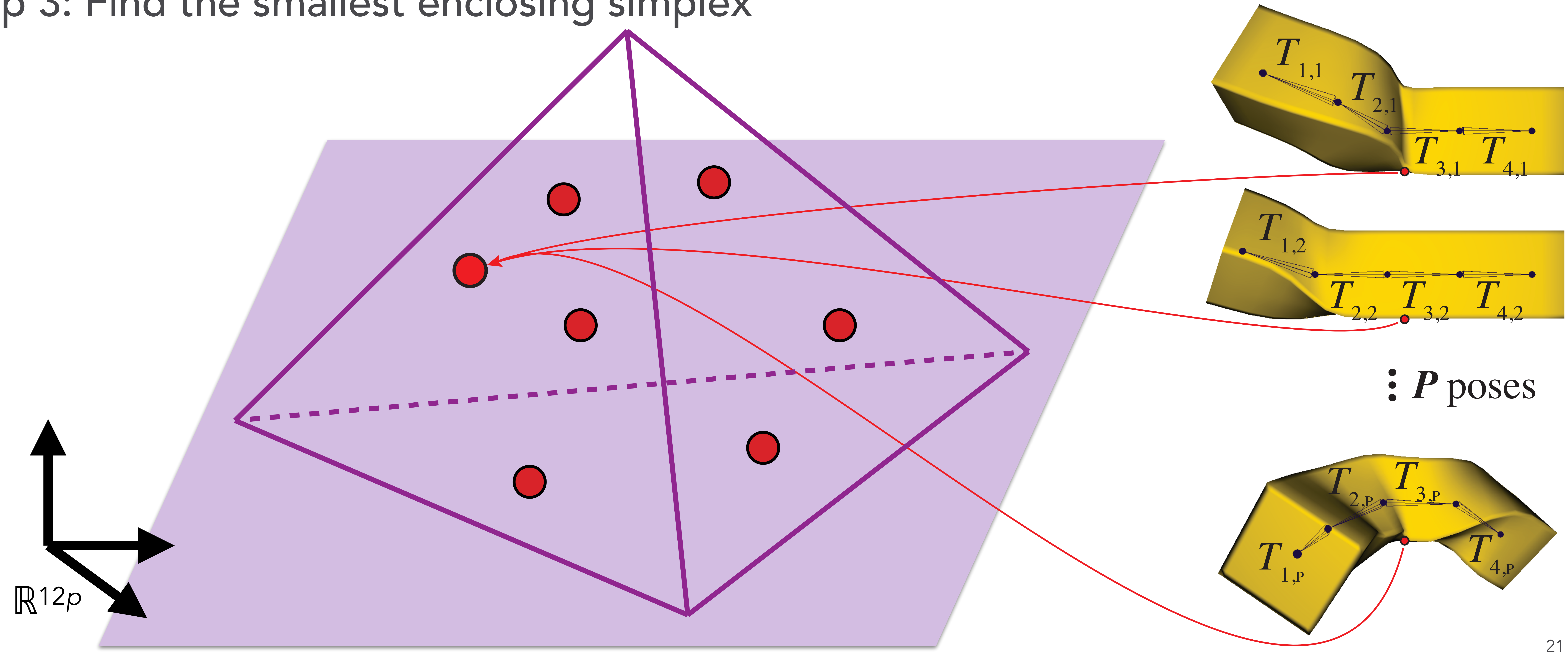
Our Approach

- Step 1: Estimate vertex transformations in $\mathbb{R}^{12p}$
- Step 2: Estimate a *#handles*-dimensional subspace for the vertices
- Step 3: Find the smallest enclosing simplex



Our Approach

- Step 1: Estimate vertex transformations in $\mathbb{R}^{12p}$
- Step 2: Estimate a #*handles*-dimensional subspace for the vertices
- Step 3: Find the smallest enclosing simplex



Hyperspectral Image Unmixing

[European Union, Copernicus Sentinel-2 imagery]

Hyperspectral Image Unmixing

- Satellites capture high-dimensional data from far away

[European Union, Copernicus Sentinel-2 imagery]

Hyperspectral Image Unmixing

- Satellites capture high-dimensional data from far away
- Pixels contain mixtures of objects

[European Union, Copernicus Sentinel-2 imagery]

Hyperspectral Image Unmixing

- Satellites capture high-dimensional data from far away
- Pixels contain mixtures of objects
- What are the objects (*endmembers*)?

[European Union, Copernicus Sentinel-2 imagery]

Hyperspectral Image Unmixing

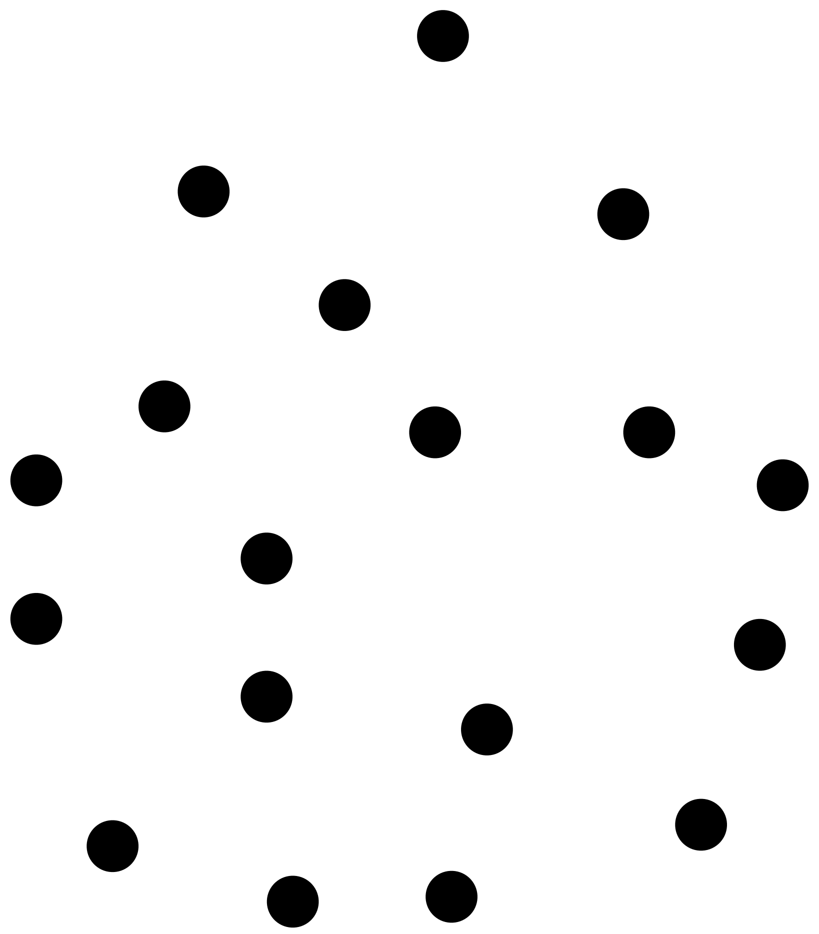
- Satellites capture high-dimensional data from far away
- Pixels contain mixtures of objects
- What are the objects (*endmembers*)?
- What mixture is in a pixel (*abundances*)?

[European Union, Copernicus Sentinel-2 imagery]

Minimum Volume Enclosing Simplex (MVES)

[Craig 1994]

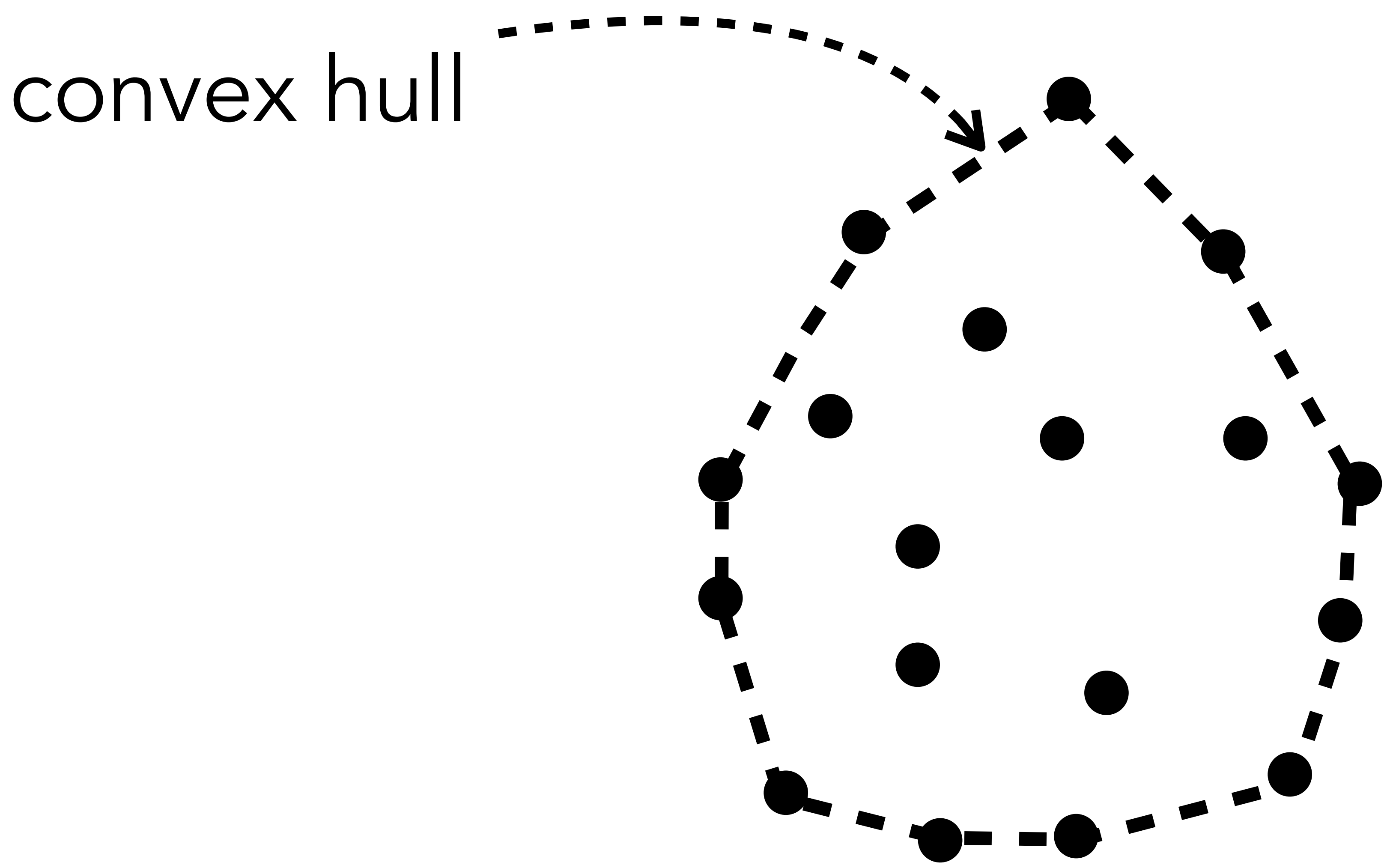
- Given points in high dimensions, perform PCA and then find the MVES



Minimum Volume Enclosing Simplex (MVES)

[Craig 1994]

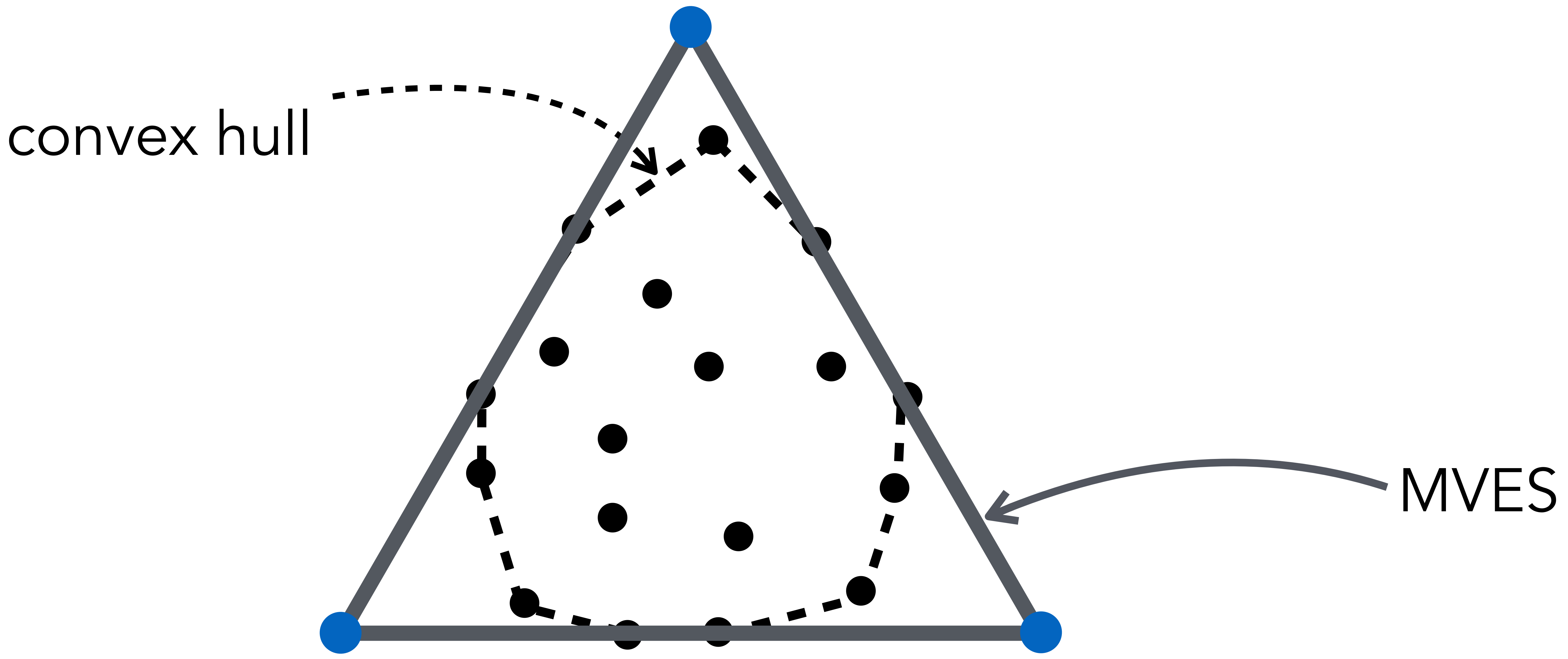
- Given points in high dimensions, perform PCA and then find the MVES



Minimum Volume Enclosing Simplex (MVES)

[Craig 1994]

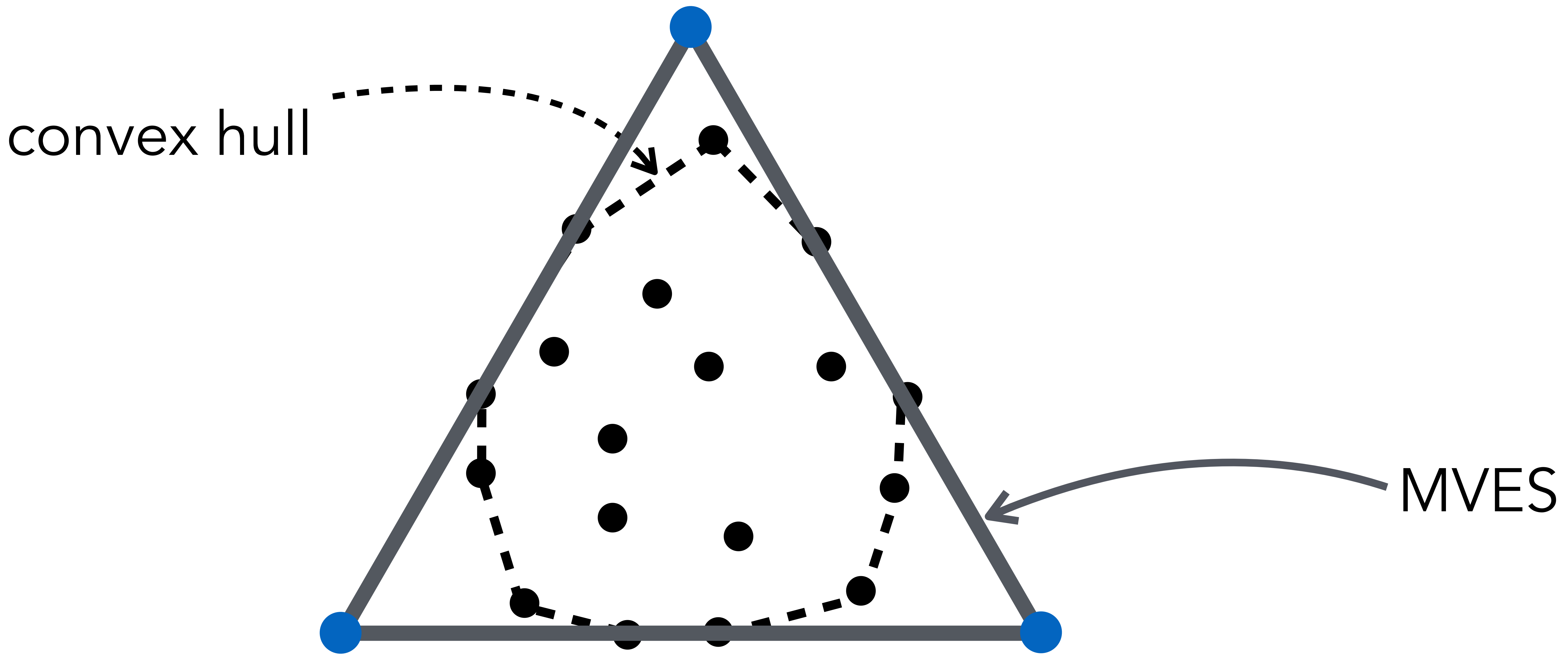
- Given points in high dimensions, perform PCA and then find the MVES



Minimum Volume Enclosing Simplex (MVES)

[Craig 1994]

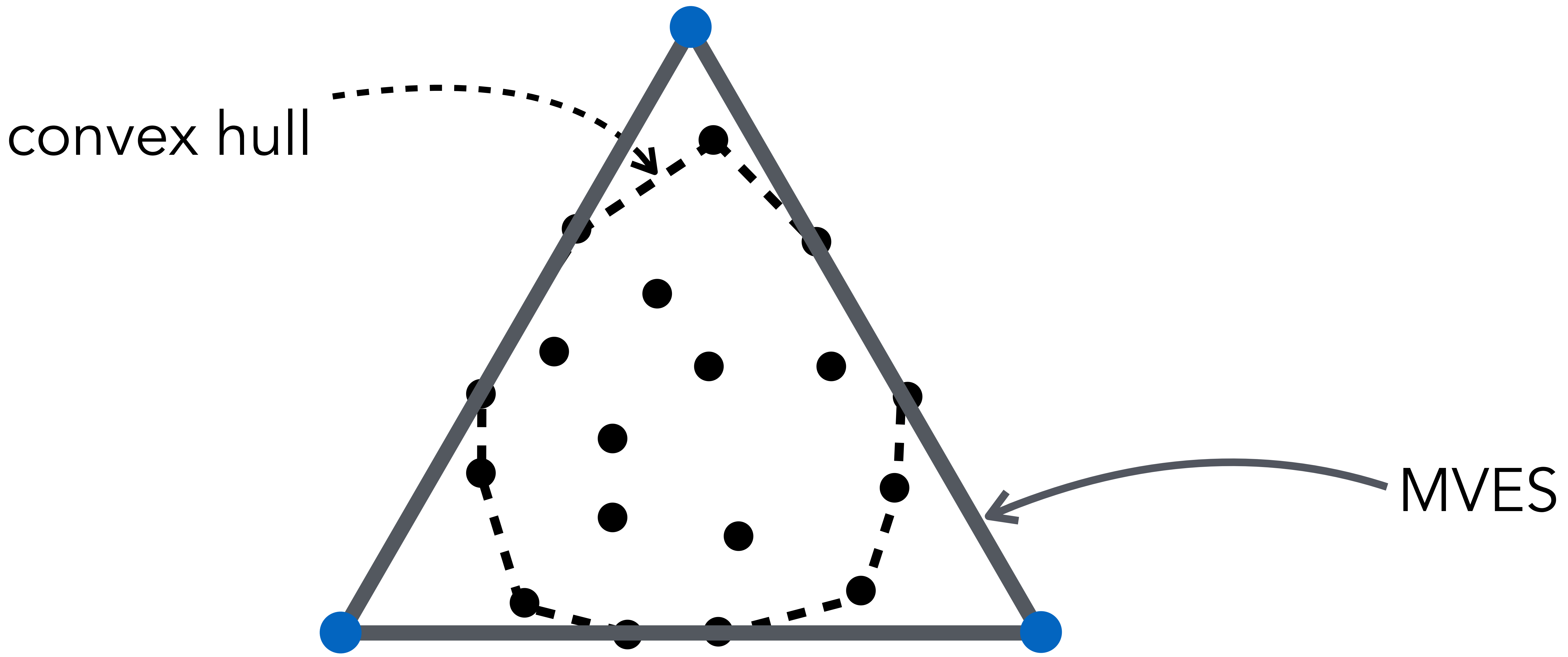
- **Given points in high dimensions, perform PCA and then find the MVES**
- **State of the art:** [Chan et al. 2009, Bioucas-Dias 2009, Ambikapathi et al. 2010, Agathos et al. 2014, Lin et al. 2016]



Minimum Volume Enclosing Simplex (MVES)

[Craig 1994]

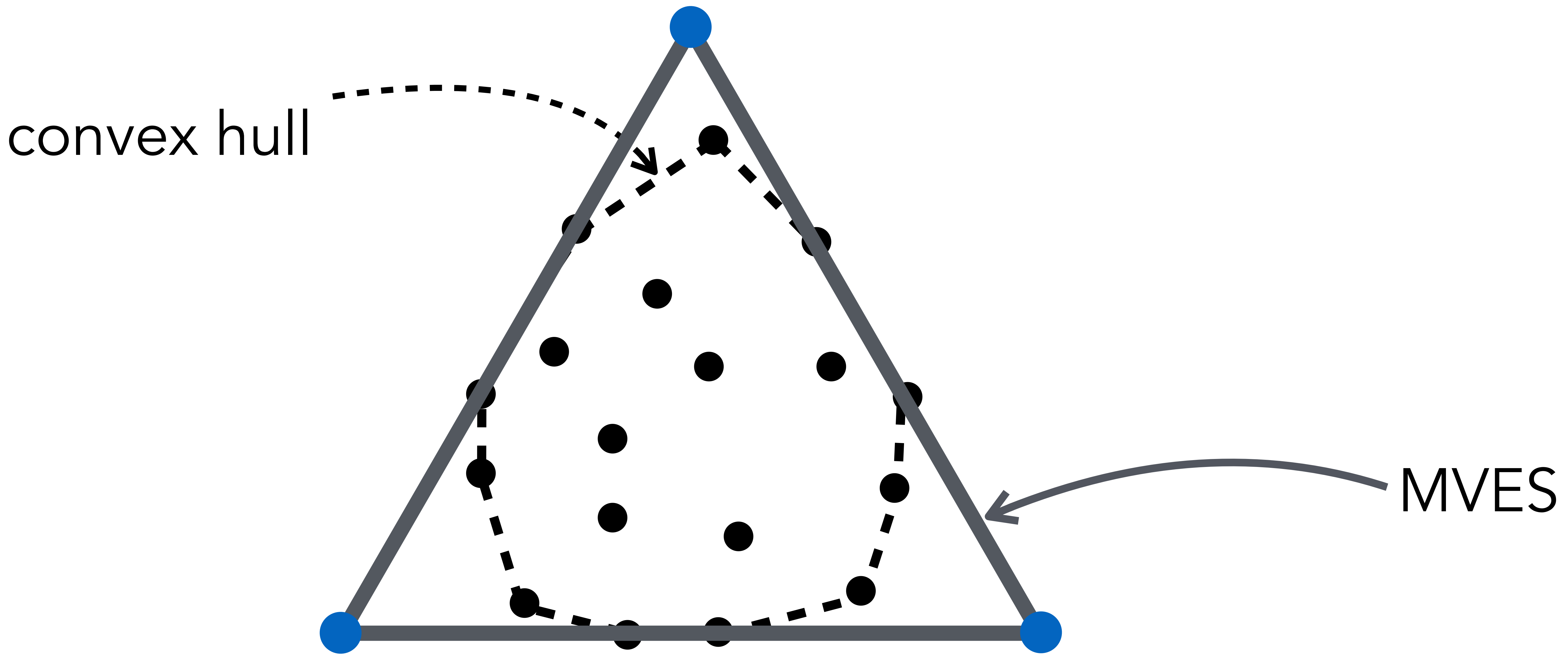
- **Given points in high dimensions, perform PCA and then find the MVES**
- **State of the art:** [Chan et al. 2009, Bioucas-Dias 2009, Ambikapathi et al. 2010, Agathos et al. 2014, Lin et al. 2016]
- **In theory, should be difficult** [Hendrix et al. 2013] **but works well in papers**



Minimum Volume Enclosing Simplex (MVES)

[Craig 1994]

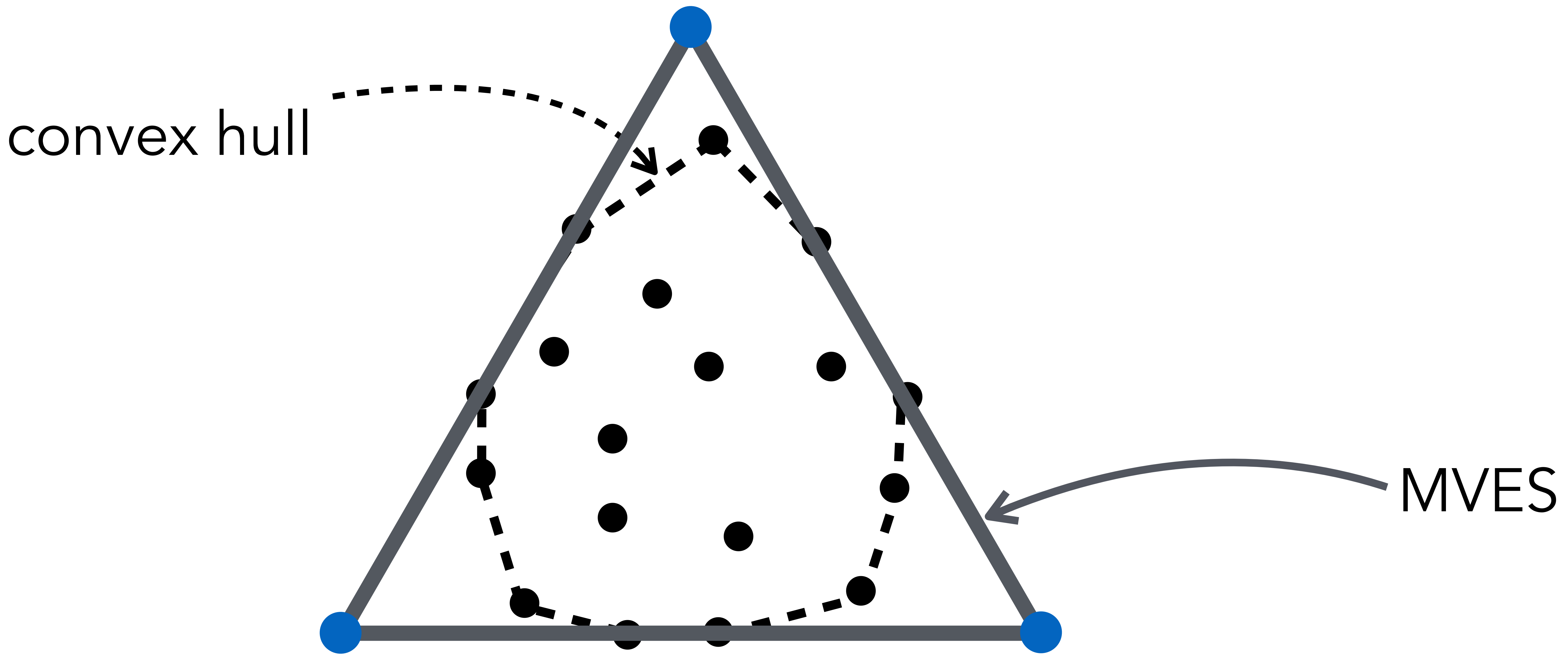
- Given points in high dimensions, perform PCA and then find the MVES
- **State of the art:** [Chan et al. 2009, Bioucas-Dias 2009, Ambikapathi et al. 2010, Agathos et al. 2014, Lin et al. 2016]
- **In theory, should be difficult** [Hendrix et al. 2013] **but works well in papers**
- **In theory, gets easier the higher the dimension** [Lin et al. 2015, Fu et al. 2015]



Minimum Volume Enclosing Simplex (MVES)

[Craig 1994]

- **Given points in high dimensions, perform PCA and then find the MVES**
- **State of the art:** [Chan et al. 2009, Bioucas-Dias 2009, Ambikapathi et al. 2010, Agathos et al. 2014, Lin et al. 2016]
- **In theory, should be difficult** [Hendrix et al. 2013] **but works well in papers**
- **In theory, gets easier the higher the dimension** [Lin et al. 2015, Fu et al. 2015]
- **Related to non-negative matrix factorization** [Arora et al. 2012]



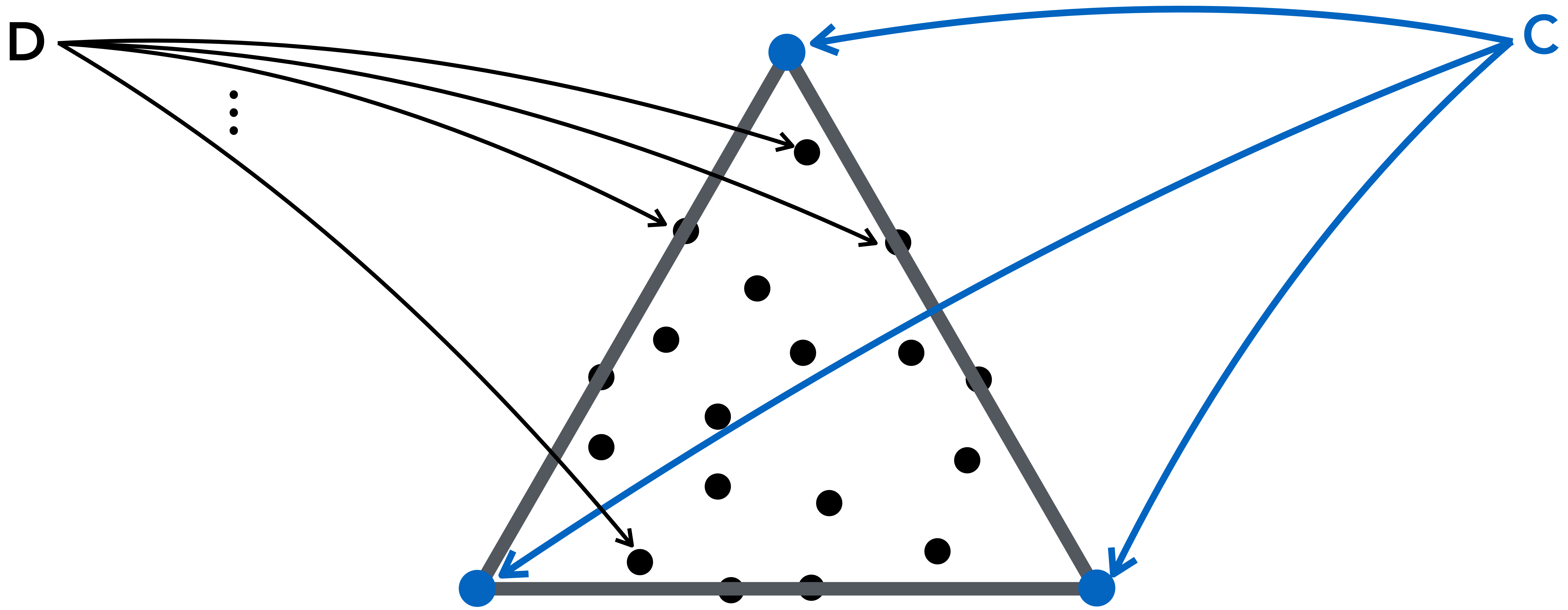
Minimum Volume Enclosing Simplex (MVES)

- Formally: $$\min_{C} |\det(C)|$$

subject to:

$$C^{-1}D \geq 0 \qquad \text{(weights ≥ 0)}$$

$$C_{h,i} = 1, \quad \forall i \in [1, h] \qquad \text{(homogeneous coordinates)}$$



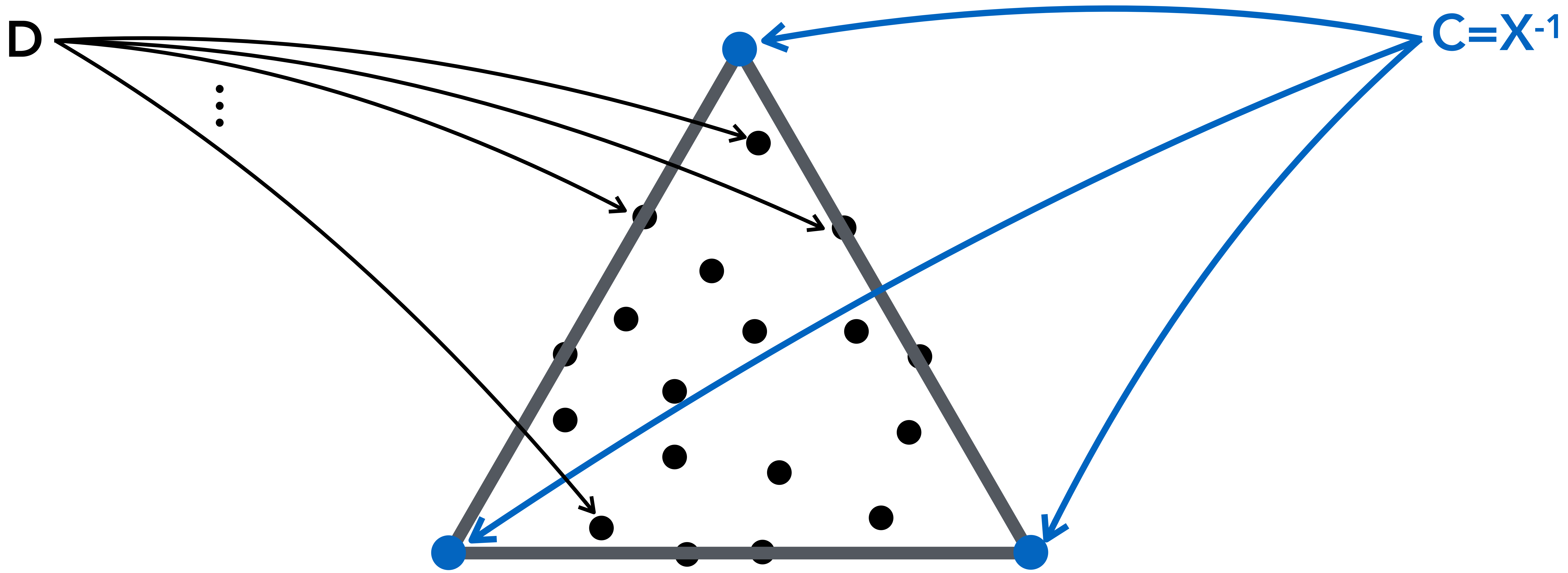
Minimum Volume Enclosing Simplex (MVES)

- Formally: $$\min(-\log\det(X))$$

subject to:

$$XD \geq 0$$ (weights ≥ 0)

$$X\mathbf{1}_h = [0,0,0,\ldots,1]^T$$ (homogeneous coordinates)



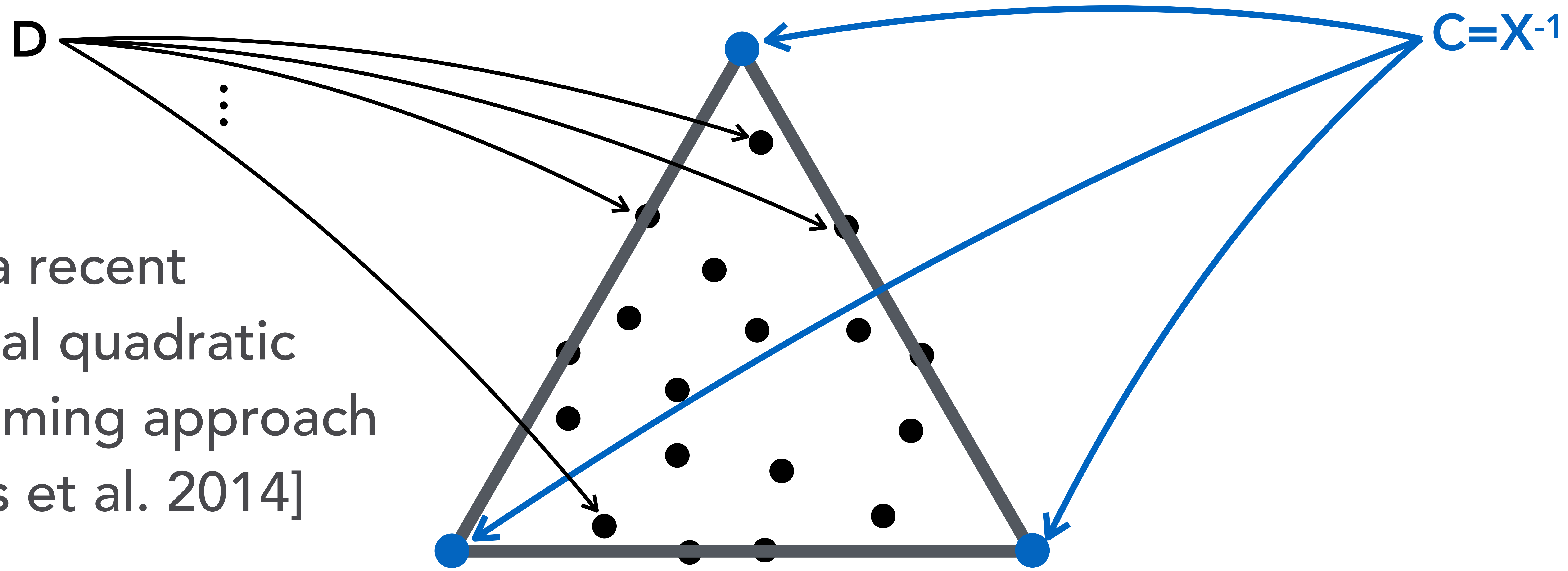
Minimum Volume Enclosing Simplex (MVES)

- Formally: $\min(-\log\det(X))$

  subject to:

  $XD \geq 0$ (weights ≥ 0)

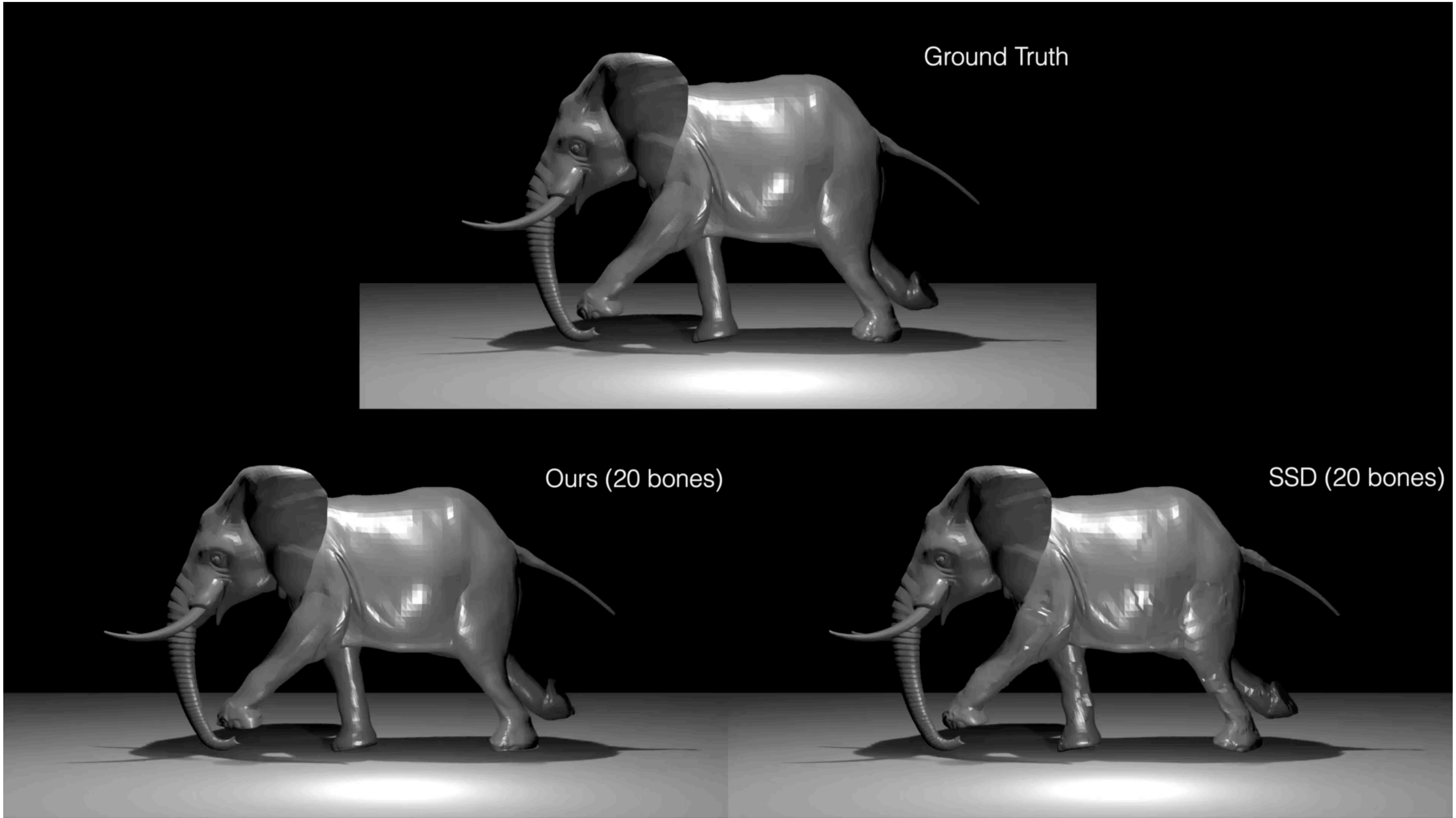
  $X\mathbf{1}_h = [0,0,0,\ldots,1]^T$ (homogeneous coordinates)



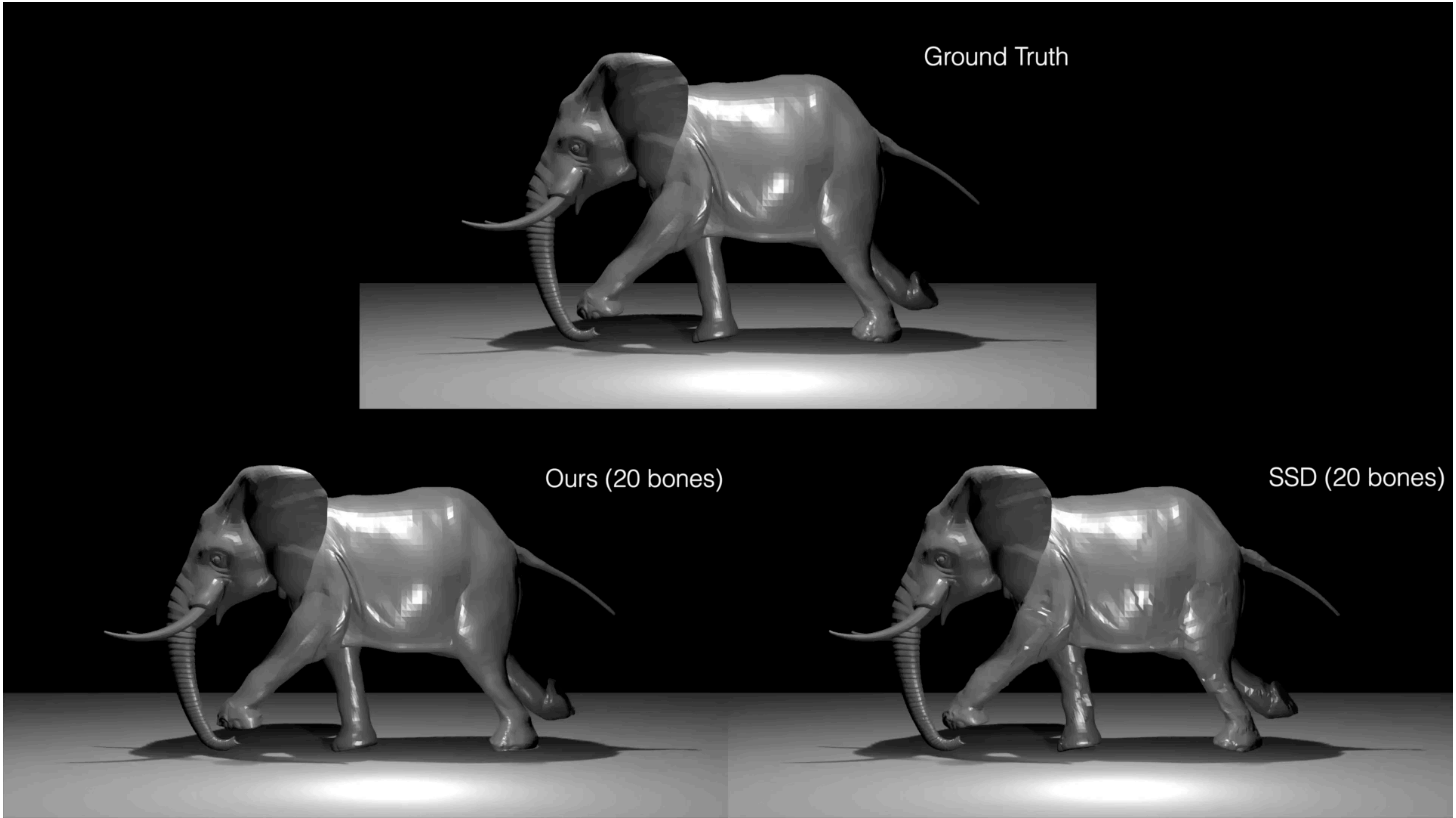
- We use a recent sequential quadratic programming approach [Agathos et al. 2014]

Results

Comparison to SSDR [Le and Deng 2012]



Comparison to SSDR [Le and Deng 2012]



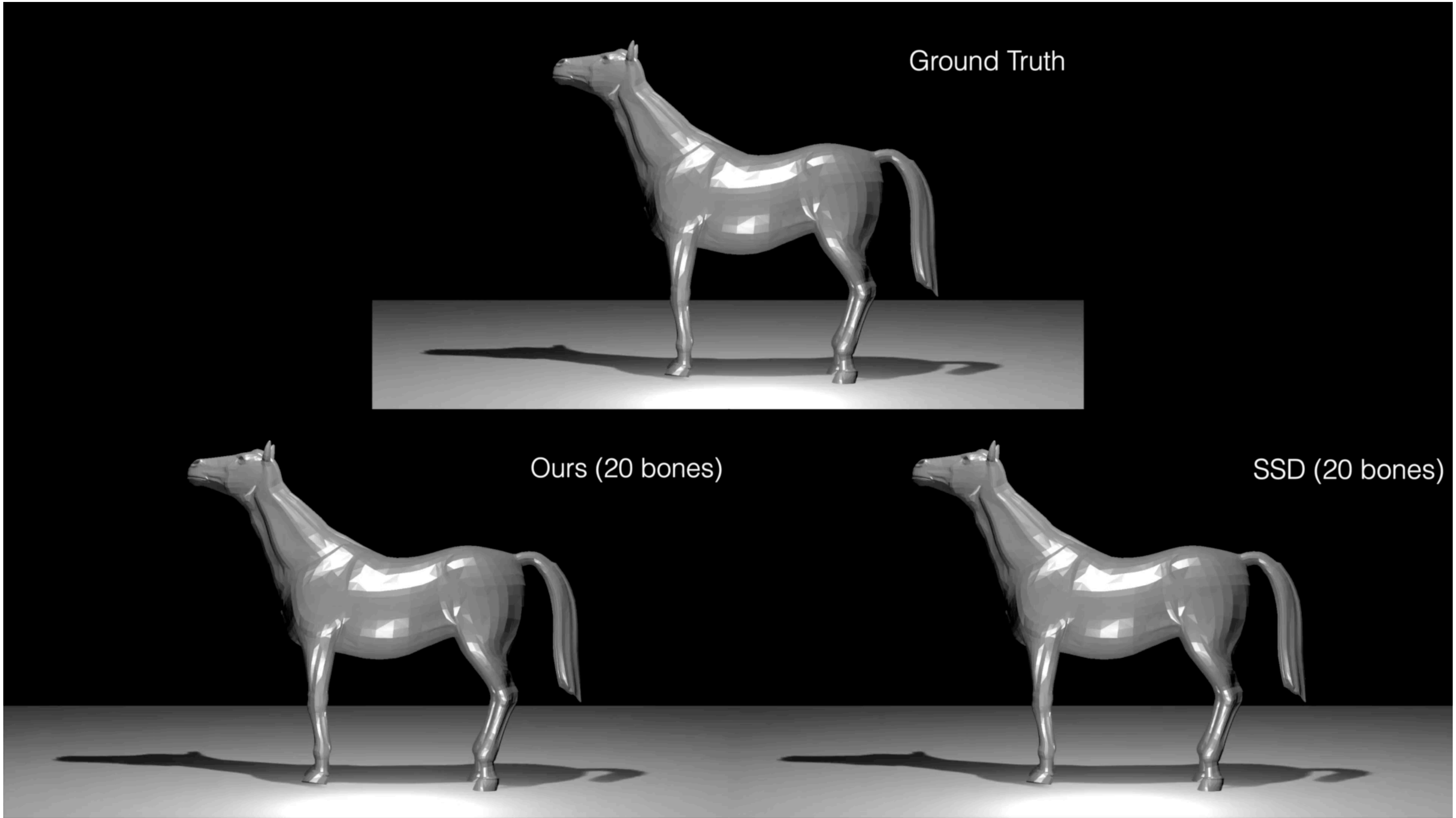
Comparison to SSDR [Le and Deng 2012]

Input	Ours	SSDR

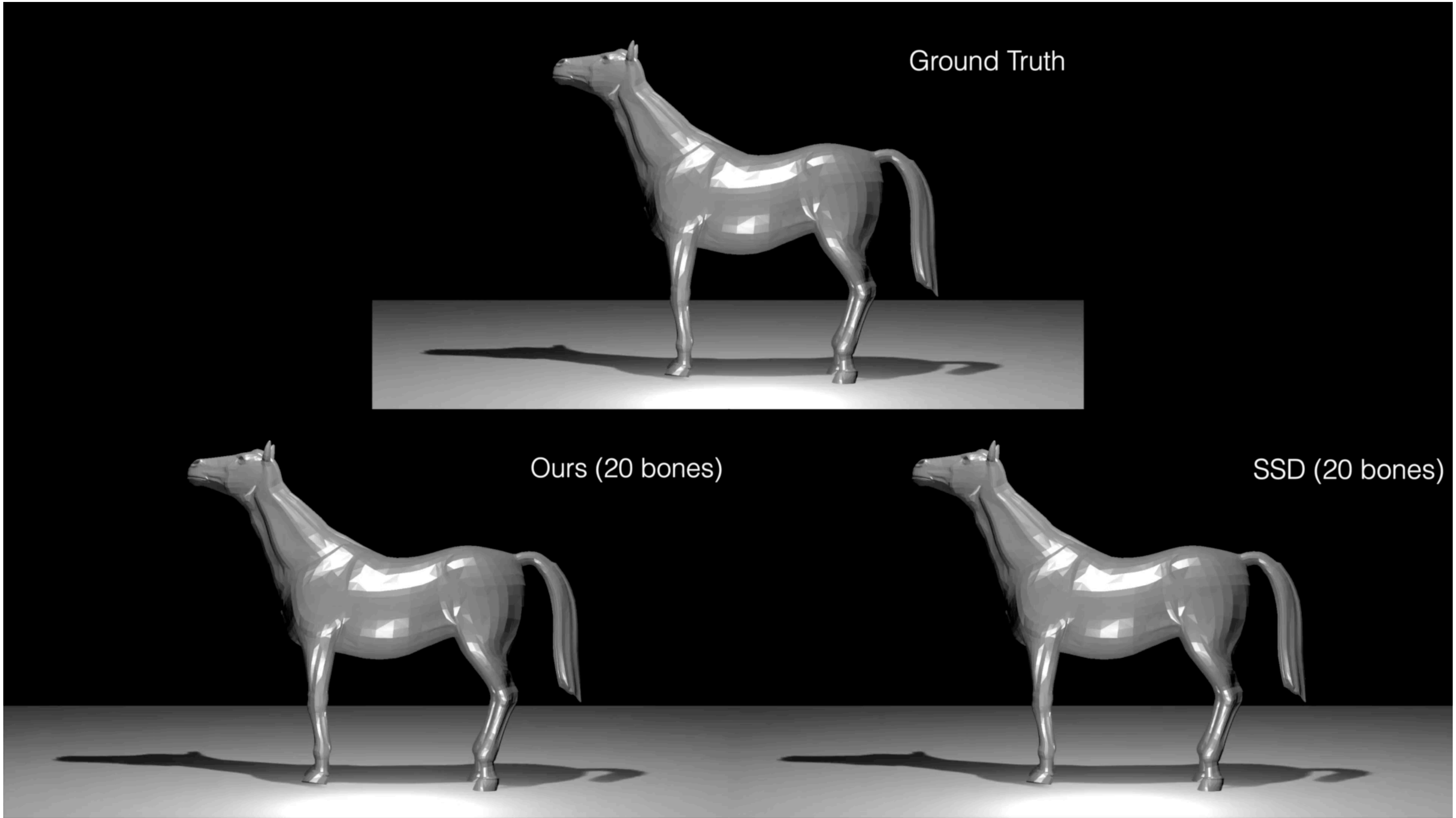
Comparison to SSDR [Le and Deng 2012]

Comparison to SSDR [Le and Deng 2012]

Comparison to SSDR [Le and Deng 2012]



Comparison to SSDR [Le and Deng 2012]



Comparison to SSDR [Le and Deng 2012]

Input Ours SSDR

Comparison to SSDR [Le and Deng 2012]

Comparison to SSDR [Le and Deng 2012]

Comparison to Kavan et al. [2010]

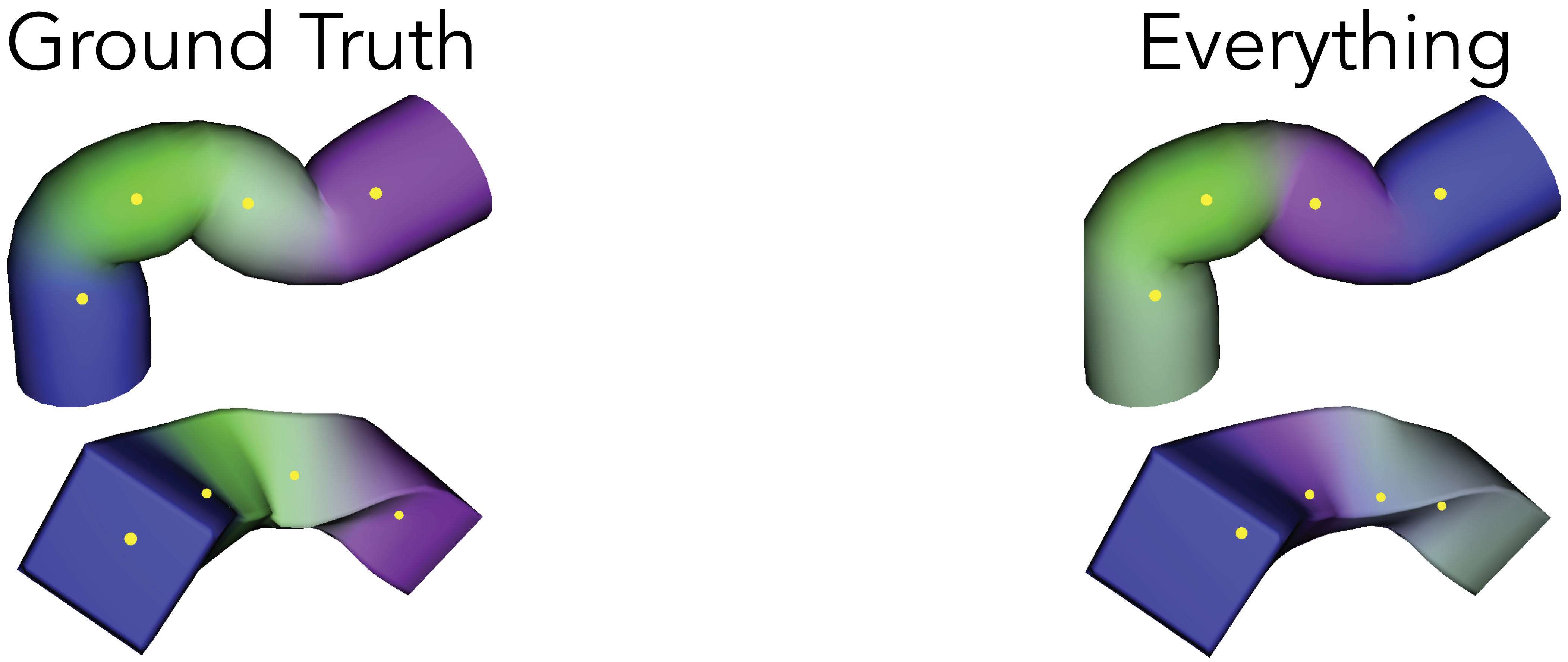
Dataset	# vertices	# poses	# bones	Approx. error E_{RMS}		Execution time (minutes)	
				Kavan et al.	Ours	Kavan et al.	Ours
crane	10002	175	40	1.4	0.73	0.36	2.66
elasticCow	2904	204	18	3.6	3.23	0.08	1.16
elephant	42321	48	25	1.4	0.46	0.37	3.49
horse	8431	48	30	1.3	0.35	0.07	0.67
samba	9971	175	30	1.5	0.86	0.26	2.1

Comparison to Kavan et al. [2010]

Dataset	# vertices	# poses	# bones	Approx. error E_{RMS}		Execution time (minutes)	
				Kavan et al.	Ours	Kavan et al.	Ours
crane	10002	175	40	1.4	0.73	0.36	2.66
elasticCow	2904	204	18	3.6	3.23	0.08	1.16
elephant	42321	48	25	1.4	0.46	0.37	3.49
horse	8431	48	30	1.3	0.35	0.07	0.67
samba	9971	175	30	1.5	0.86	0.26	2.1

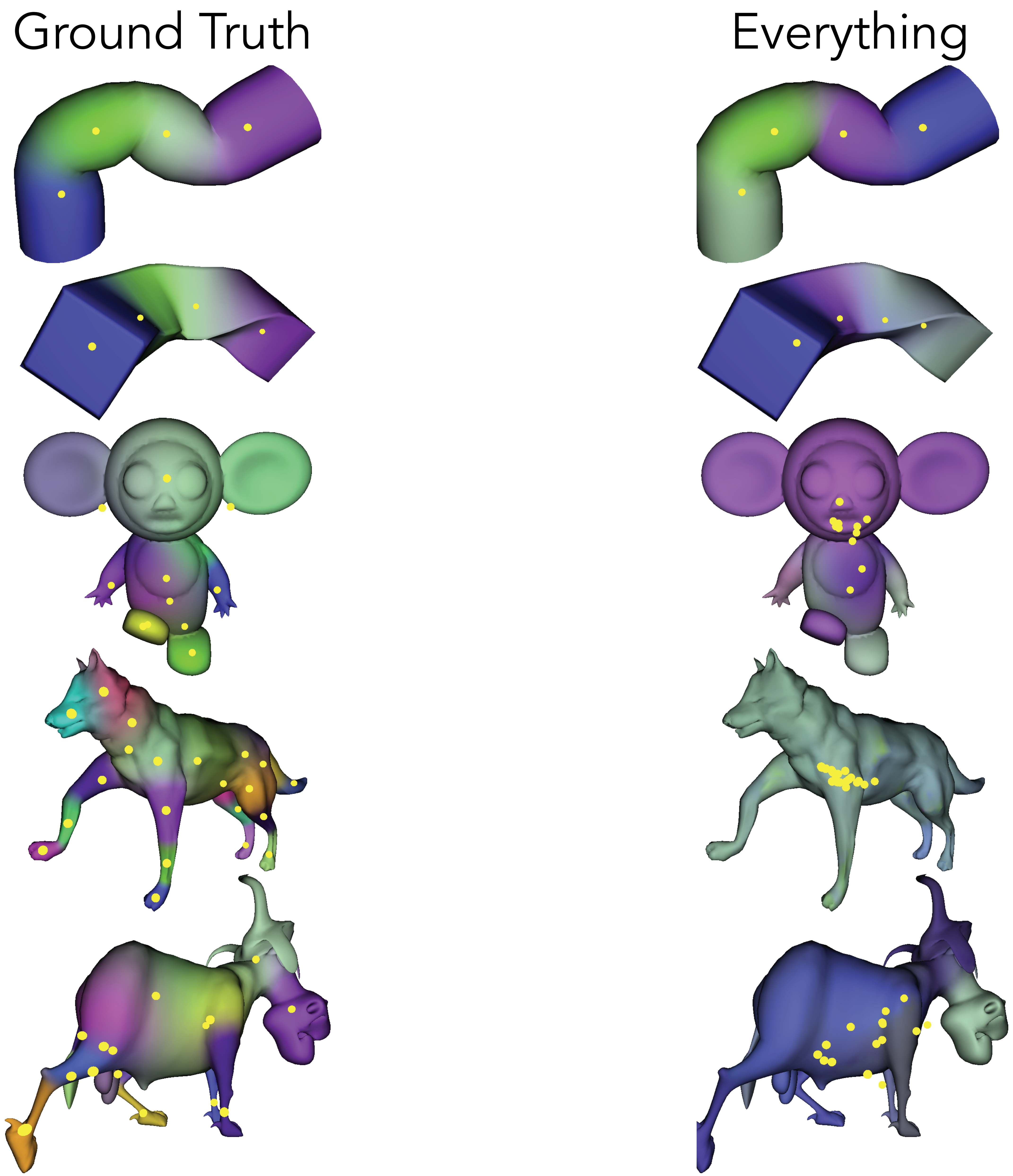
Recovering Ground Truth

- Our approach recovers ground truth for simple cases



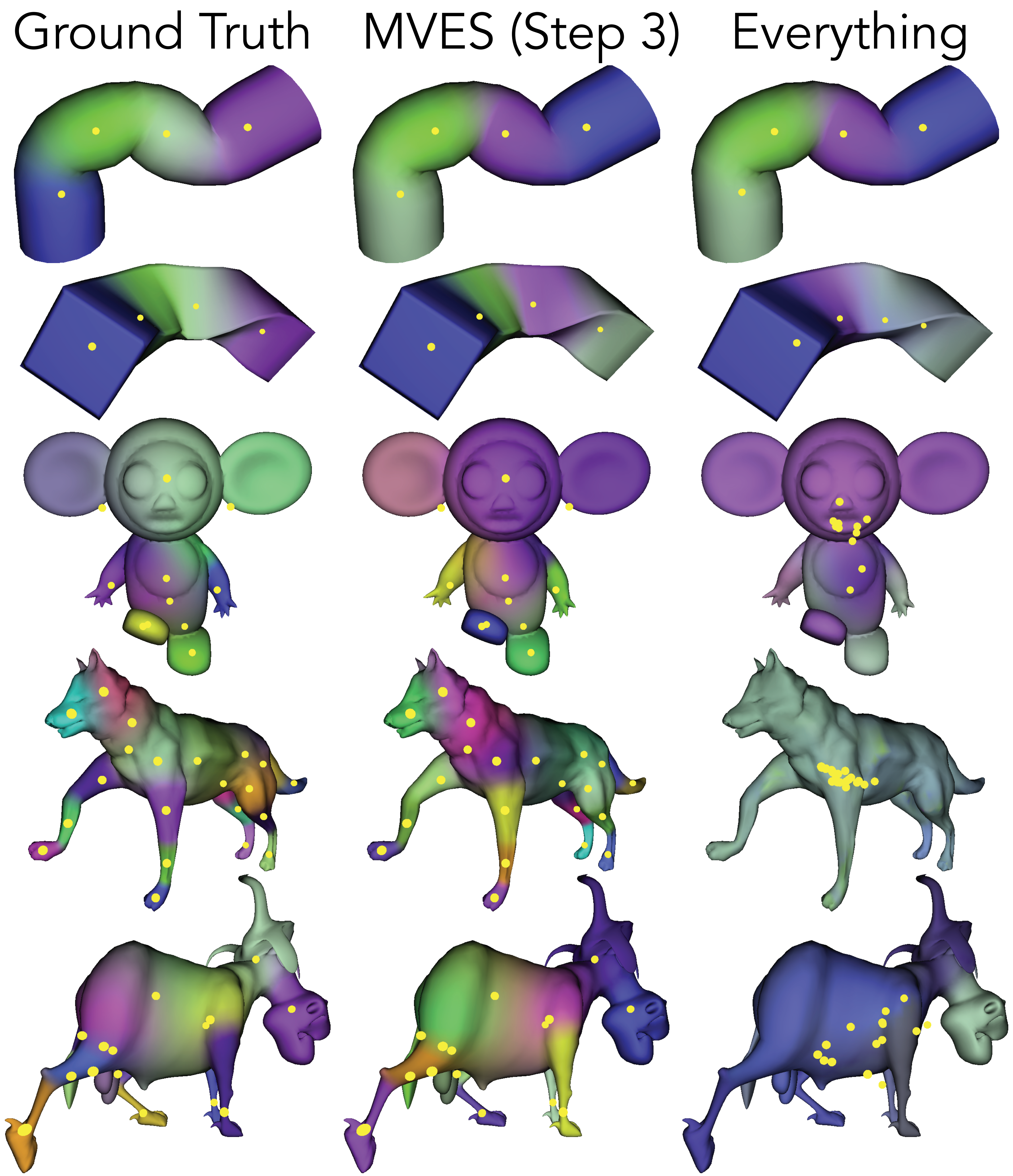
Recovering Ground Truth

- Our approach recovers ground truth for simple cases
- Always recovers vertex positions (perhaps with different handle transformations and weights)

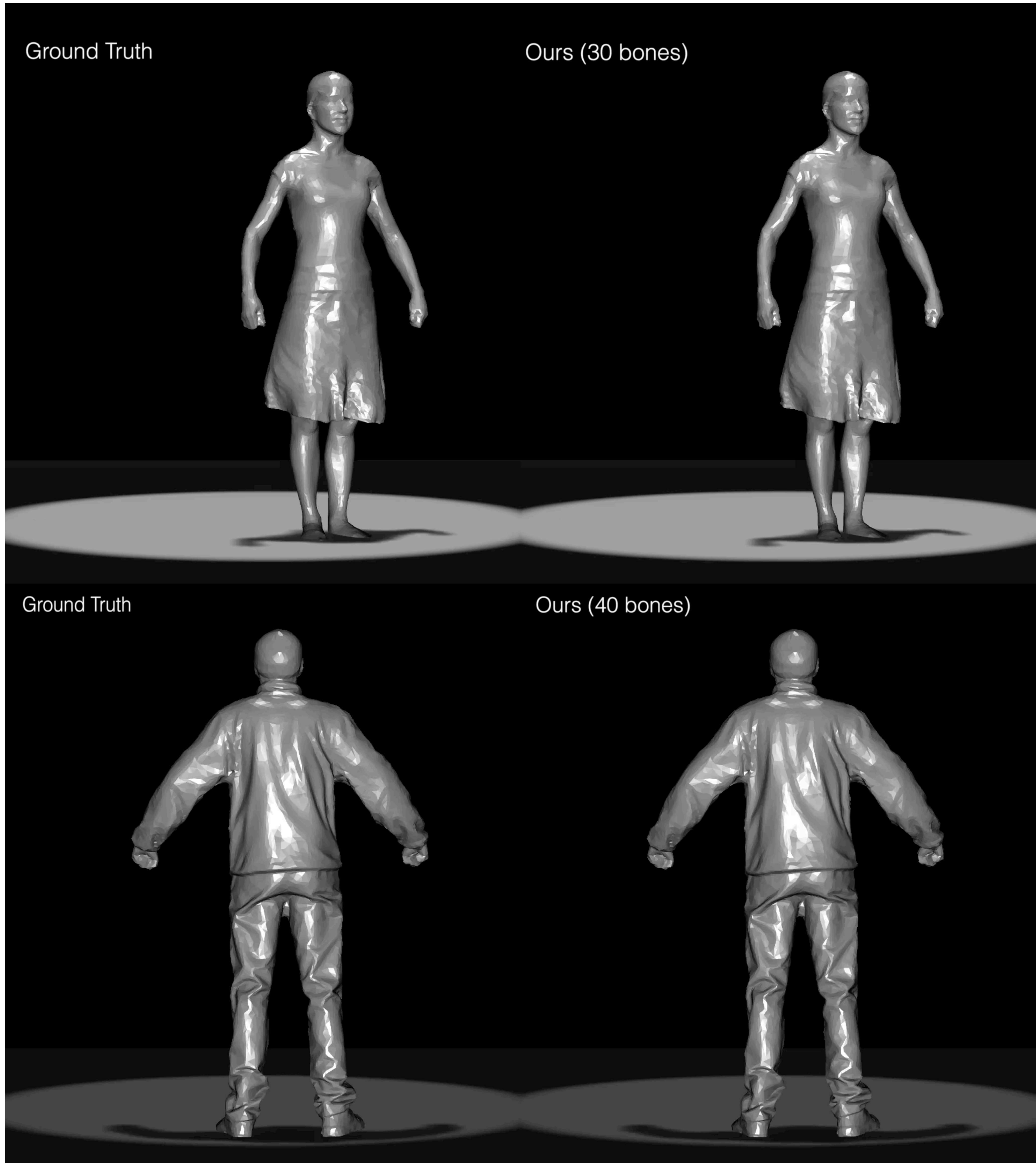


Recovering Ground Truth

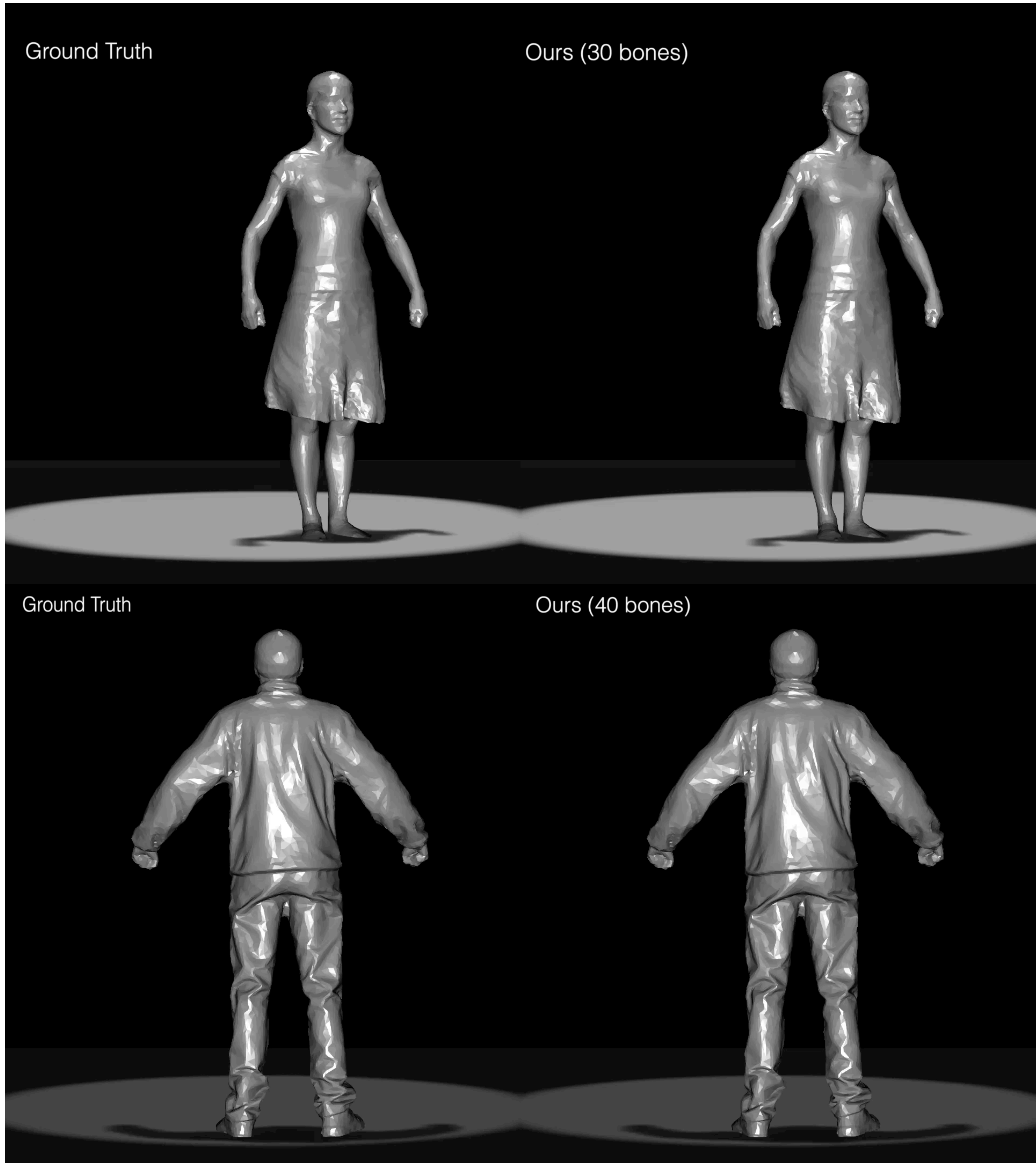
- Our approach recovers ground truth for simple cases
- Always recovers vertex positions (perhaps with different handle transformations and weights)
- Given true per-vertex transformations, MVES recovers true handles and weights



Mesh Animation Compression

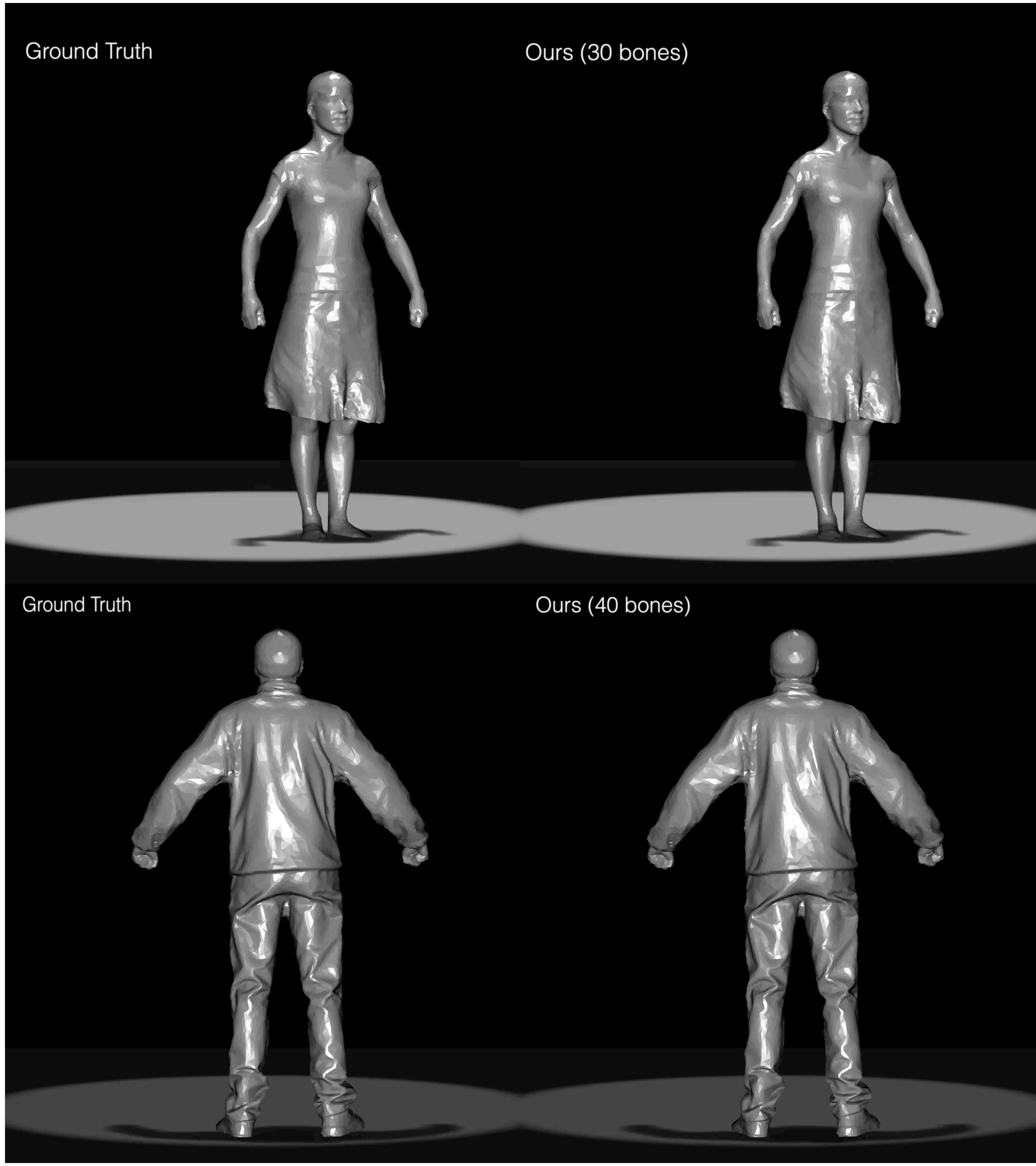


Mesh Animation Compression



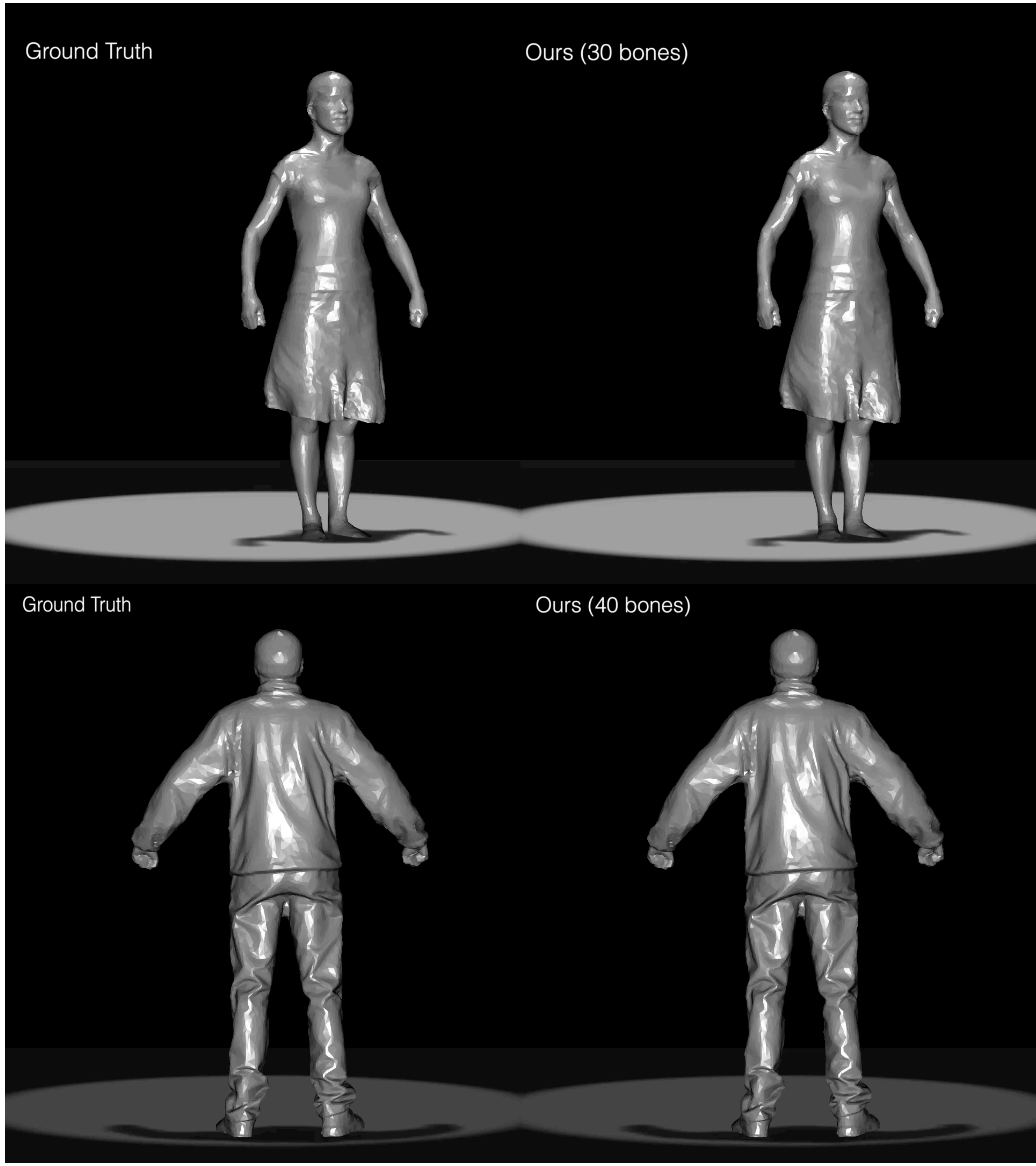
Mesh Animation Compression

- Measured in bits per vertex per frame (bpfv)



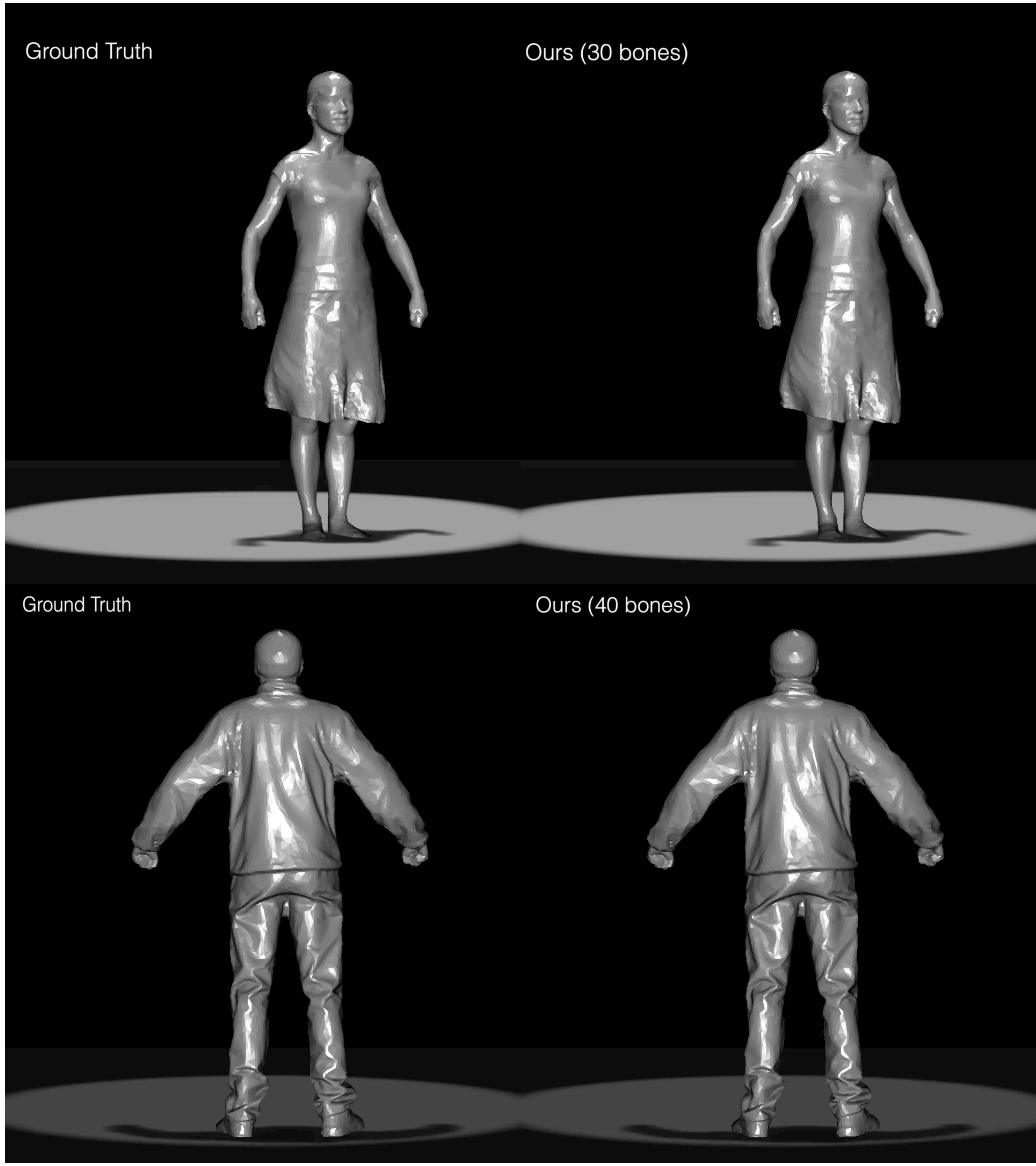
Mesh Animation Compression

- Measured in bits per vertex per frame (bpfv)
- Weights are a one-time per-vertex cost
  - $32h$ bits per vertex (h floats/vertex · 32 bits/float)



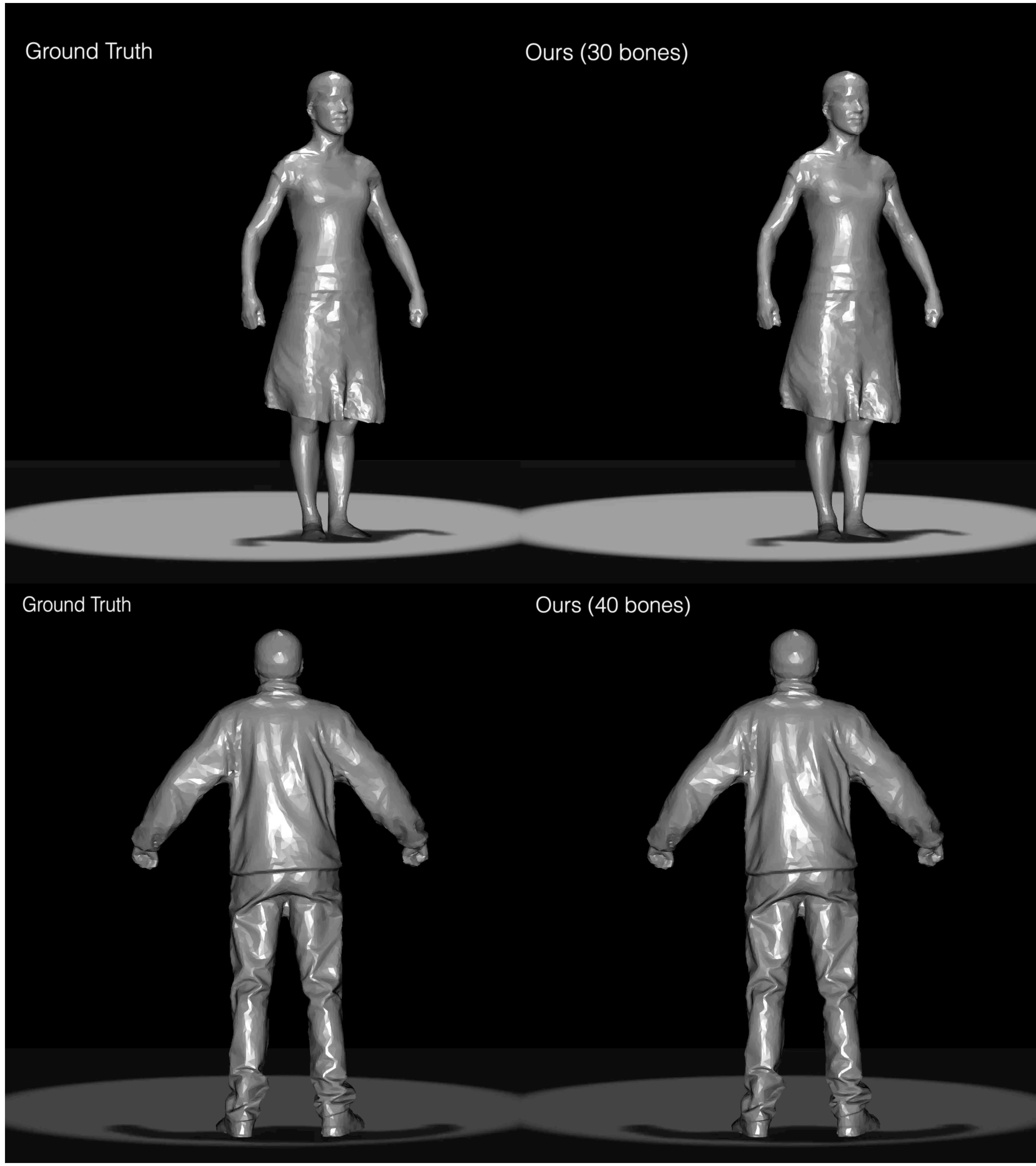
Mesh Animation Compression

- Measured in bits per vertex per frame (bpfv)
- Weights are a one-time per-vertex cost
  - $32h$ bits per vertex (h floats/vertex · 32 bits/float)
- Each frame: one affine matrix per *handle,* shared by all vertices
  - bpfv = $12h$/#vertices · 32 bits
    (12 floats/handle · 32 bits/float amortized over all vertices)
  - very low incremental cost per frame

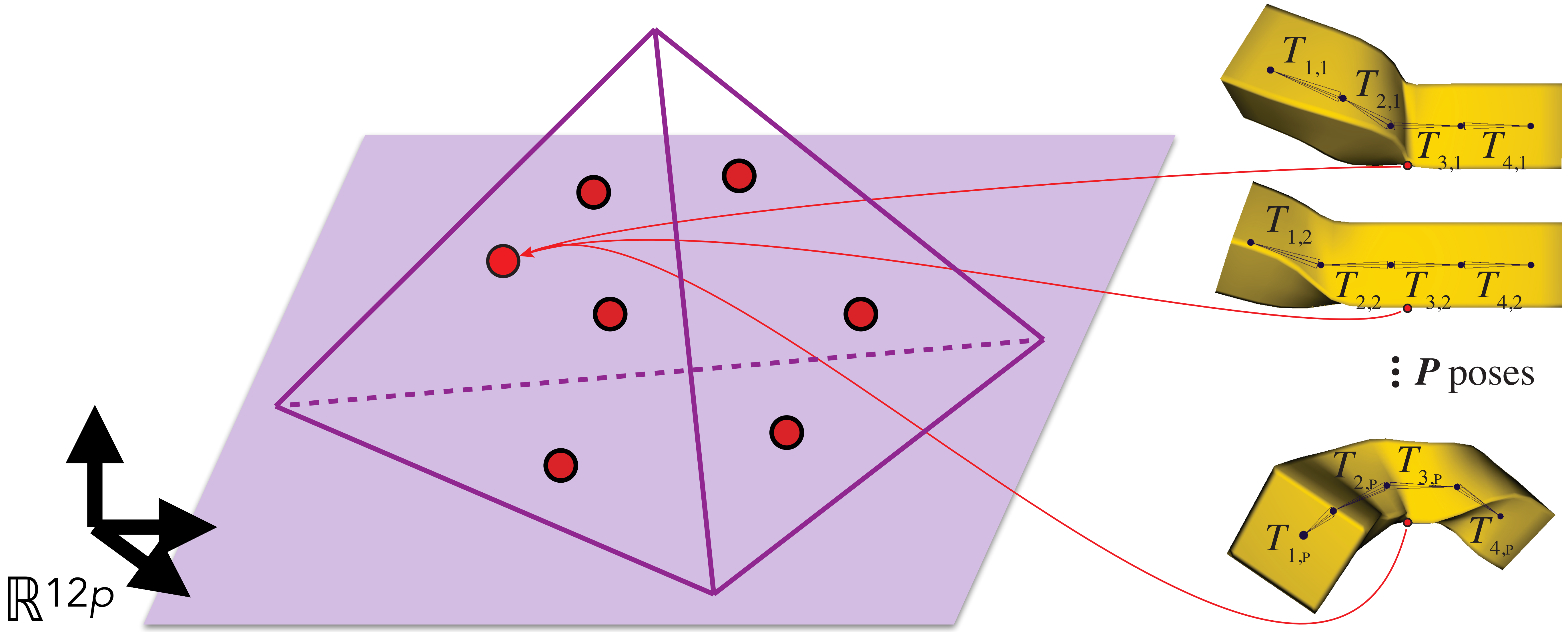


Mesh Animation Compression

- Measured in bits per vertex per frame (bpfv)
- Weights are a one-time per-vertex cost
  - $32h$ bits per vertex (h floats/vertex · 32 bits/float)
- Each frame: one affine matrix per *handle,* shared by all vertices
  - bpfv = $12h$/#vertices · 32 bits
    (12 floats/handle · 32 bits/float amortized over all vertices)
  - very low incremental cost per frame
- 4.6× lower error than state of the art [Luo et al. 2019]



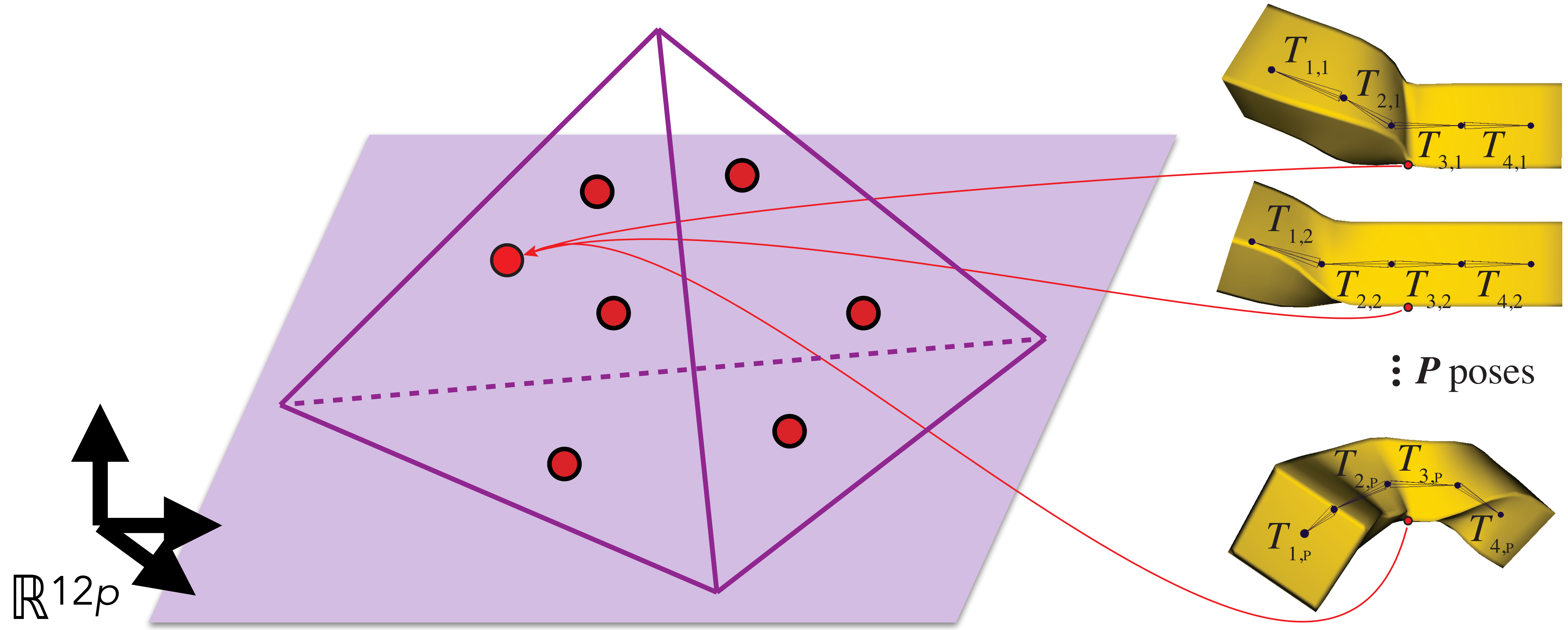
Conclusion



$\mathbb{R}^{12p}$
$T_{1,1}$
$T_{2,1}$
$T_{3,1}$
$T_{4,1}$
$T_{1,2}$
$T_{2,2}$
$T_{3,2}$
$T_{4,2}$
P poses
$T_{1,P}$
$T_{2,P}$
$T_{3,P}$
$T_{4,P}$

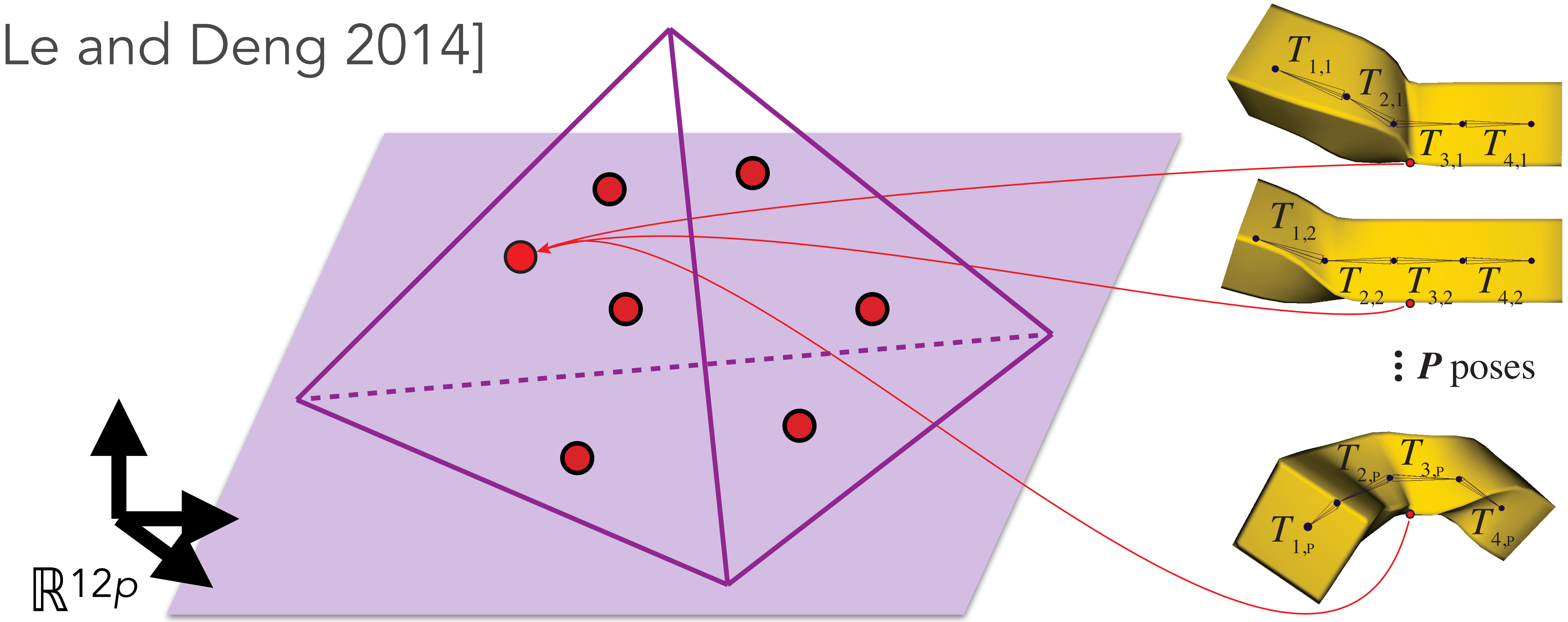
Conclusion

- Inverse Skinning is a problem in high-dimensional geometry
  - Simple expression
  - Benefits from improvements in Hyperspectral Image Unmixing
  - Benefits from improvements to the closest flat problem



Conclusion

- Inverse Skinning is a problem in high-dimensional geometry
  - Simple expression
  - Benefits from improvements in Hyperspectral Image Unmixing
  - Benefits from improvements to the closest flat problem
- Limitations
  - Transformations aren't rigid. They makes them less useful when editing.
  - No sparsity. Sometimes LBS weights aren't sparse, but this is often desirable.
  - We don't recover a bone skeleton [Le and Deng 2014]



Thank You

- Code and data: https://cragl.cs.gmu.edu/hyperskinning/

- Acknowledgements:
  - Harry Gingold, Alec Jacobson, Shahar Kovalsky, Arthur Dupre, Jie Gao, and Leonard Schulman for informative discussions
  - Kyle Falicov for help with rendering
  - Guoliang Luo for running his algorithm on our data
- Financial support
  - US NSF (IIS-1524782 & IIS-1453018)
  - Google
  - Adobe