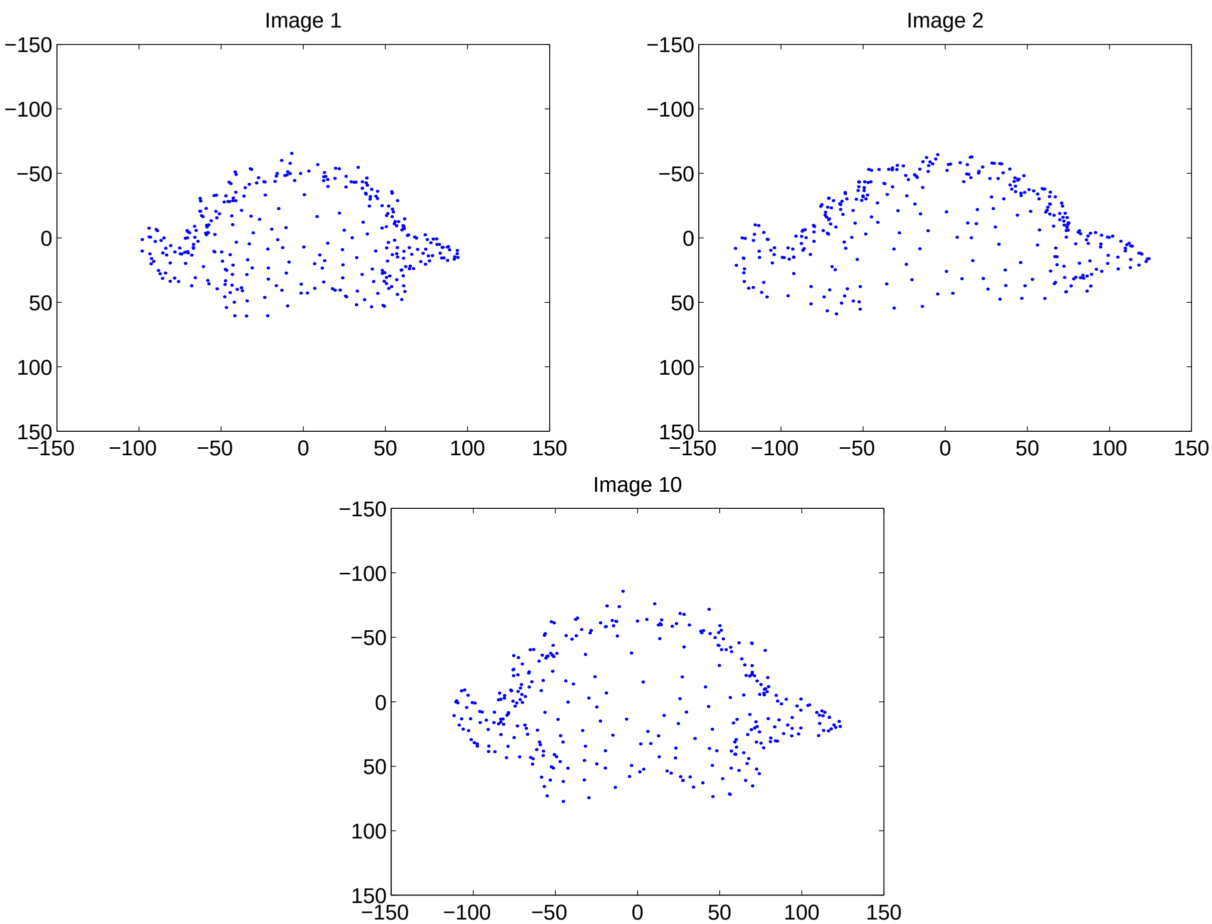
Multi-Frame Factorization Techniques

Suppose $\{\vec{x}_{j,n}\}_{j=1,n=1}^{J,N}$ is a set of *corresponding* image coordinates, where the index $n = 1, \ldots, N$ refers to the n^{th} scene point and $j = 1, \ldots, J$ refers to the j^{th} image.



Such corresponding points may be obtained from local feature points, for example.

Problem. Estimate the 3D point positions, $\{\vec{X}_n\}_{n=1}^{N}$, along with the placement and calibration parameters for the J cameras.

Perspective Projection

The image points and the scene points are related by perspective projection,

$$\vec{p}_{j,n} = \frac{1}{z_{j,n}} M_j \vec{P}_n. \tag{1}$$

Here $\vec{p}_{j,n} = (x_{j,n}, y_{j,n}, 1)^T$ is in homogeneous pixel coordinates, and the scene point $\vec{P}_n = (P_{n,1}, P_{n,2}, P_{n,3}, 1)^T$ is in homogeneous 3D coordinates. Also $M_j = M_{in,j} M_{ex,j}$ is the 3×4 camera matrix formed from the product of the intrinsic and extrinsic calibration matrices. Finally, $z_{j,n}$ is the projective depth, $z_{j,n} = \vec{e}_3^{\,T} M_j \vec{P}_n$, where $\vec{e}_3^{\,T} = (0, 0, 1)$ (i.e. $\vec{e}_3$ is the third standard unit vector).

For convenience we assume the intrinsic matrices have the form

$$M_{in,j} = \begin{pmatrix} f_j & 0 & 0 \\ 0 & f_j & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{2}$$

The extrinsic calibration matrices are in general given by

$$M_{ex,j} = \left(R_j, -R_j \vec{d}_j \right), \tag{3}$$

where R_j is the rotation from the world to the j^{th}-camera's coordinates, and $\vec{d}_j$ is the position, in world coordinates, of the nodal point for the j^{th} camera.

Bundle Adjustment

We wish to solve for the point positions $\vec{P}_n$ for $n = 1, \ldots, N$ and the camera matrices M_j for $j = 1, \ldots, J$ by minimizing

$$\mathcal{O}(\{M_j\}_{j=1}^{J}, \{\vec{P}_n\}_{n=1}^{N}) \equiv \sum_{j,n} \left\| \begin{pmatrix} x_{j,n} \\ y_{j,n} \end{pmatrix} - \frac{1}{\vec{e}_3^T M_j \vec{P}_n} \left(I_2,\ \vec{0}\right) M_j \vec{P}_n \right\|^2 . \tag{4}$$

Here the camera matrices M_j must be of the form $M_{in,j} M_{ex,j}$ where $M_{in,j}$ and $M_{ex,j}$ are as given in equations (2) and (3).

This nonlinear optimization problem is called **bundle adjustment**.

In these notes we discuss two approximations to bundle adjustment:

1. Approximate perspective projection by scaled orthographic projection.
2. Rescale each term in the bundle adjustment objective function (4) and solve a bilinear problem.

Scaled-Orthographic Projection

Scaled-orthographic projection provides an approximation of perspective projection (1) for the case of narrow fields of view,

$$\max\{|x_{j,n}|, |y_{j,n}|\} << f_j,$$

and relatively shallow depth variations,

$$z_{j,n} \approx 1/s.$$

For scaled-orthographic projection, the image points and the scene points are related by

$$\left(I_2,\ \vec{0}\right) \vec{p}_{j,n} = s \left(I_2,\ \vec{0}\right) M_j \vec{P}_n. \tag{5}$$

Here $\vec{p}_{j,n}$, $\vec{P}_n$ and M_j are as above, and s is a constant scale factor. This is **bilinear** in the scaled camera matrix sM_j and the 3D point $\vec{P}_n$.

Differences from Mean Image Points

Let $\bar{\vec{p}}_j = \frac{1}{N}\Sigma_{n=1}^{N}\vec{p}_{j,n}$ be the average image point, and $\bar{\vec{P}} = \frac{1}{N}\Sigma_{n=1}^{N}\vec{P}_n$ be the average scene point. Then, by equation (5), we can show

$$\vec{d}_{j,n} = \tilde{M}_j \vec{D}_n, \tag{6}$$

where

$$\begin{aligned}
\vec{d}_{j,n} &= \left(I_2,\ \vec{0}\right)(\vec{p}_{j,n} - \bar{\vec{p}}_j),\\
\vec{D}_n &= (I_3,\ \vec{0})(\vec{P}_n - \bar{\vec{P}}),\\
\tilde{M}_j &= s\left(I_2,\ \vec{0}\right) M_j \left(I_3,\ \vec{0}\right)^T.
\end{aligned}$$

Moreover, from equations (2) and (3) it follows that the scaled-orthographic projection matrix $\tilde{M}_j$ has the form

$$\tilde{M}_j = s\begin{pmatrix} f_j & 0 & 0 \\ 0 & f_j & 0 \end{pmatrix} R_j = s f_j \left(I_2,\ \vec{0}\right) R_j, \tag{7}$$

where R_j is the rotation matrix for the j^{th} camera, as above.

Derivation: Difference from Mean

Let $\vec{\bar{p}}_j = \frac{1}{N}\sum_{n=1}^{N} \vec{p}_{j,n}$ be the average image point in the j^{th} image, and $\vec{\bar{P}} = \frac{1}{N}\sum_{n=1}^{N} \vec{P}_n$ be the average scene point.

Then, by equation (5), we have

$$\left(I_2,\ \vec{0}\right)\vec{\bar{p}}_j = s\left(I_2,\ \vec{0}\right)M_j\vec{\bar{P}}$$

Subtracting this from (5) we find

$$\left(I_2,\ \vec{0}\right)(\vec{p}_{j,n} - \vec{\bar{p}}_j) = s\left(I_2,\ \vec{0}\right)M_j(\vec{P}_n - \vec{\bar{P}}).$$

Note the 4^{th} component of $\vec{P}_n - \vec{\bar{P}}$ is equal to $1 - 1 = 0$. Therefore we can drop this 4^{th} component, and obtain

$$\vec{d}_{j,n} = \tilde{M}_j\vec{D}_n,$$

where

$$\begin{aligned}
\vec{d}_{j,n} &= \left(I_2,\ \vec{0}\right)(\vec{p}_{j,n} - \vec{\bar{p}}_j),\\
\vec{D}_n &= (I_3,\ \vec{0})(\vec{P}_n - \vec{\bar{P}}),\\
\tilde{M}_j &= s\left(I_2,\ \vec{0}\right)M_j\left(I_3,\ \vec{0}\right)^T.
\end{aligned}$$

Which is what we set out to show.

Notice we can use the definitions of $M_{in,j}$ and $M_{ex,j}$ to simplify $\tilde{M}_j$ above. We find

$$\begin{aligned}
\tilde{M}_j &= s\left(I_2,\ \vec{0}\right)M_j\left(I_3,\ \vec{0}\right)^T,\\
&= s\left(I_2,\ \vec{0}\right)M_{in,j}M_{ex,j}\left(I_3,\ \vec{0}\right)^T,\\
&= s\begin{pmatrix} f_j & 0 & 0 \\ 0 & f_j & 0 \end{pmatrix}R_j
\end{aligned}$$

This gives equation (7) above.

Scaled-Orthographic Factorization

Let $C = \left(\vec{d}_{j,n}\right)$ be the $2J \times N$ data matrix formed by stacking $\vec{d}_{j,n}$, for $j = 1, \ldots, J$ in columns, and combining these columns for $n = 1, \ldots, N$ (here j is the camera index and n the feature point index). From above, $\vec{d}_{j,n} = \vec{x}_{j,n} - \bar{\vec{x}}_j$, where $\vec{x}_{j,n}$ is the observed pixel position of the j^{th} point in the n^{th} frame, and $\bar{\vec{x}}_j$ is the average of these over all n. In particular, the data matrix can be built from the observed corresponding points.

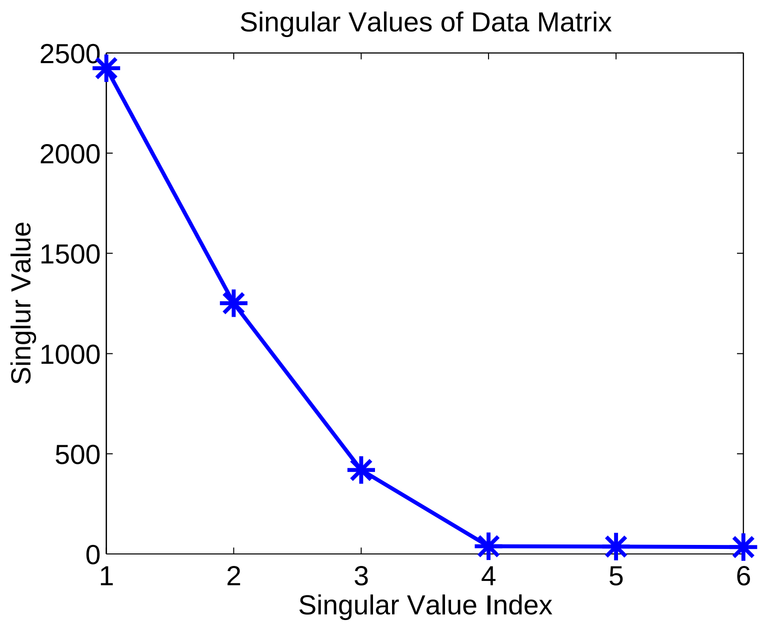
From equation (6) we then have

$$C = MD, \tag{8}$$

where M is the $2J \times 3$ matrix formed by stacking the $\tilde{M}_j$ matrices, and D is the $3 \times N$ shape matrix having columns given by $\vec{D}_n$. This equation states that **the data matrix has at most rank 3** (without considering noise).

Factorization via SVD

Performing an SVD on the data matrix C, for a case with $J = 3$ images, provides $C = W\Sigma V^T$ with the singular values shown below:



See the 3dRecon Matlab tutorial `orthoMassageDino.m` ($\sigma_n = 1$ pixel).

Affine Shape

What does the factorization $C = W\Sigma V^T$ tell us about the shape of the objects being imaged? For notational convenience, we assume that all but the first 3 singular values of Σ have been set to zero or, equivalently, Σ is 3×3, W is $2J \times 3$ and V^T is $3 \times N$.

We now have two rank 3 factorizations of C, namely MD and $W\Sigma V^T$. But this factorization is only unique up to a nonsingular matrix A, as follows:

$$C = MD = (WA)(A^{-1}\Sigma V^T) = W\Sigma V^T. \tag{9}$$

That is, for some 3×3 matrix A, the 3D point positions and the camera matrices must be given by

$$\begin{aligned} D &= A^{-1}\Sigma V^T \\ M &= WA. \end{aligned} \tag{10}$$

Equivalently, we could place AA^{-1} between the Σ and the V^T in equation (9).

Therefore we know the shape D up to the 9 parameters in A, namely $AD = \Sigma V^T$. This is known as an **affine reconstruction** of the shape D.

Affine Shape (Cont.)

What can A (or, equivalenty, A^{-1}) do to a shape?

For example, consider a configuration of 3D points as specified by the $3 \times N$ matrix D above. Suppose we have a nonsingular matrix A. What does the configuration AD look like?

Use SVD to decompose A into $U_a \Sigma_a V_a^T$. So $AD = U_a(\Sigma_a(V_a^T D))$ is obtained by rotating/reflecting D using V_a^T, then stretching/shrinking the result along the axes according to Σ_a, and finally rotating/reflecting this result using U_a. (Imagine applying such transforms to your lecturer's head.)

The equivalence class of all configurations that can be obtained with transformations of this form is called **affine shape**.

It can be shown that affine shape preserves parallel lines and intersecting lines, but not angles and lengths.

See Tomasi and Kanade, IJCV, Vol. 9, 1992, pp.137-154, for the original factorization method.

Euclidean Reconstructions

We can determine many of the parameters in A from knowledge about the cameras.

In particular, suppose we know the projection matrix $\tilde{M}_j$ satisfies

$$\tilde{M}_j \;=\; sf_j \left(I_2,\ \vec{0} \right) R_j,$$

for some value of sf_j. From (10) we have $\tilde{M}_j = W_j A$ where W_j is the j^{th} 2×3 block in W corresponding to the same two rows as $\tilde{M}_j$ occupies in M. Since $R_j R_j^T = I_3$ it then follows that

$$\tilde{M}_j \tilde{M}_j^T = s^2 f_j^2 I_2 = W_j A A^T W_j^T. \tag{11}$$

Here the scale factor for the j^{th} image sf_j and the 3×3 symmetric positive definite matrix $Q = AA^T$ are the only unknowns.

For each j, equation (11) provides 2 linear homogeneous equations for the coefficients of Q. Then for $J \geq 3$ we have $2J \geq 6$ homogeneous linear equations which we can solve for Q, up to a scalar multiple r_q^2. Finally, given Q we can factor it (assuming the eigenvalues are all non-negative) by computing the eigenvalues, $Q = U_q \Lambda_q U_q^T$, and then recognizing $A = \frac{1}{r_q} U_q \Lambda_q^{1/2} R_q^T$. Here r_q is the unknown scale factor in Q, and R_q is an arbitrary orthogonal 3×3 matrix.

Euclidean Reconstruction (Cont.)

Therefore we have recovered $A = \frac{1}{r_q} K_q R_q^T$ where $K_q = U_q \Lambda_q^{1/2}$ is known. As a consequence we have recovered the shape matrix D_r and the camera matrix M_r where

$$\begin{aligned} D &= r_q R_q D_r, \quad \text{for } D_r = K_q^{-1} \Sigma V^T, \\ M &= \frac{1}{r_q} M_r R_q^T, \quad \text{for } M_r = W K_q. \end{aligned} \tag{12}$$

This is called a **Euclidean reconstruction**, since we have recovered the shape up to a 3D scale r_q, and a rotation/reflection R_q. Equivalently, this is referred to as a **metric shape** recovery.

The ambiguity of the overall rotation R_q reflects the fact that we cannot recover the orientation of the original world coordinate frame. This unknown rotation R_q affects both the shape, via $D = R_q D_r$, and all of the camera matrices, via $M = M_r R_q^T$. That is, R_q rotates the both the scene and the cameras together.

Similarly, the ambiguity of the overall scale r_q reflects the fact that we do not know the scale of the world coordinate frame. We could be imaging a tiny scene with large scale factors sf_j, and we could not tell from the images alone. (Think about making the movie Titanic.) Here r_q rescales the shape via $D = r_q D_r$, and also rescales all the scale parameters f_j in the cameras, via $M = \frac{1}{r_q} M_r$.

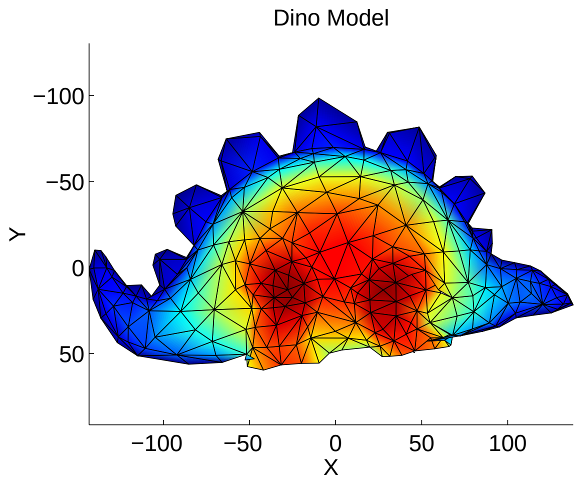
Remaining Ambiguities

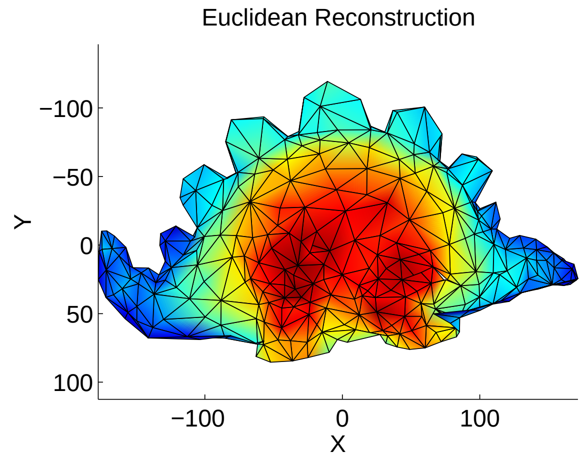
The remaining ambiguity in R_q is the **Necker ambiguity**, that is, R_q could be a reflection (say $R_q = diag(1, 1, -1)$). Effectively, with orthographic projection we cannot tell the difference between a concave-in shape viewed from the left, and the reflected concave-out shape viewed from the right. Unlike the previous two ambiguities, this ambiguity does not persist (mathematically) when we switch to perspective projection.

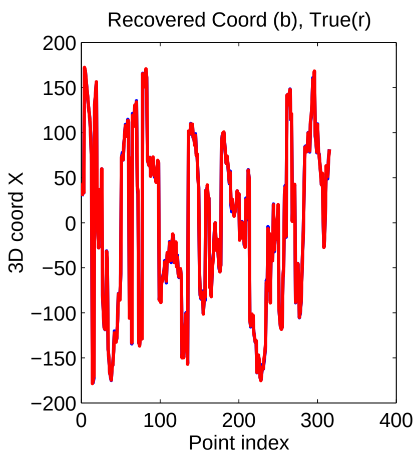
For $J = 2$ orthographic views there is an additional ambiguity, known as the **bas-relief** ambiguity. For this ambiguity, there is an additional unknown parameter (in K_q above), which ties the overall depth variation of the shape to the amount of rotation between the two cameras. See `orthoMassageDino.m`.

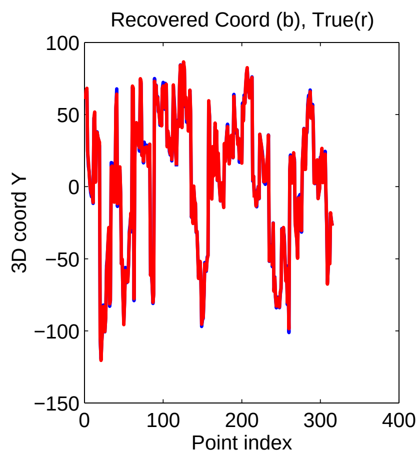
Refs: See the classic paper by Koenderink and van Doorn, 1991.

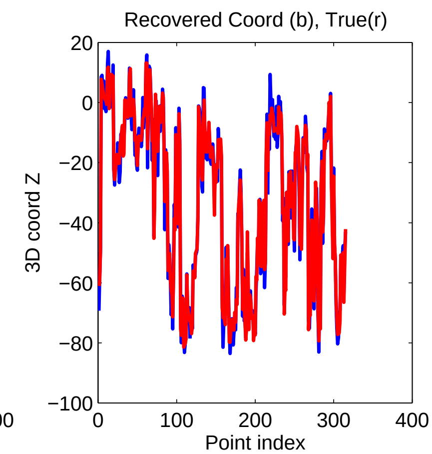
Dino Example, Orthographic Case

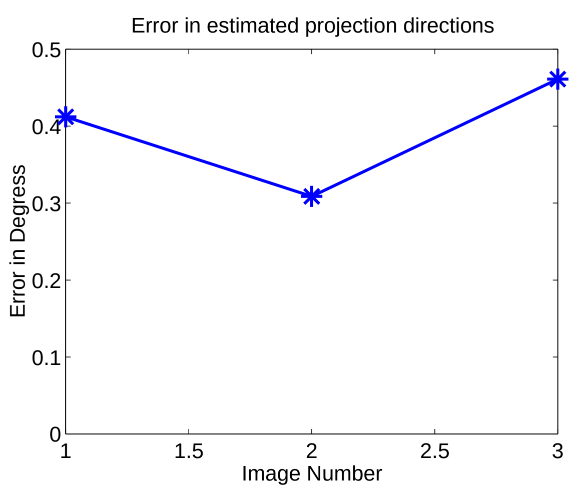


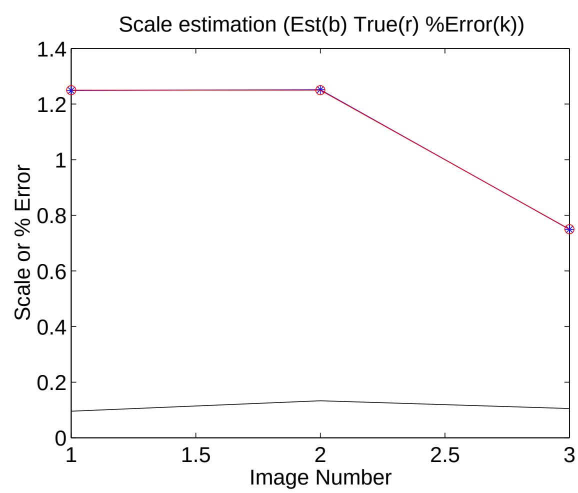












Introduction to Projective Reconstruction

Returning to perspective projection, it is tempting to modify the bundle adjustment objective function (4) by multiplying each term in the sum by the projective depths $z_{j,n} = \vec{e}_3^{\,T} M_j \vec{P}_n$, providing a reweighted version of (4):

$$\mathcal{O} = \sum_{j,n} \left\| z_{j,n} \begin{pmatrix} x_{j,n} \\ y_{j,n} \\ 1 \end{pmatrix} - M_j \vec{P}_n \right\|^2 \tag{13}$$

where $z_{j,n}$, M_j and $\vec{P}_n$ are all unknowns for $j = 1, \ldots, J$ and $n = 1, \ldots, N$.

The form of (13) suggests the following factorization approach.

Projective Factorization

Suppose we know the projective depths $z_{j,n}$, and form the data matrix $C = (z_{j,n}\vec{p}_{j,n})$. This is a $3J \times N$ matrix formed by stacking the 3-vectors $z_{j,n}\vec{p}_{j,n}$ in columns for the same point n, and then arranging these columns side by side for $n = 1, \ldots, N$. By equation (1) we have

$$z_{j,n}\vec{p}_{j,n} \;=\; M_j P_n.$$

Therefore C (for the correct $z_{j,n}$'s) must have the rank 4 factorization

$$C = MP, \tag{14}$$

where M is the $3J \times 4$ matrix formed by stacking up the camera matrices M_j, and $P = (\vec{P}_1, \ldots, \vec{P}_N)$ is the $4 \times N$ shape matrix.

Iterative Projective Factorization

Suppose we normalize $C_n = CL$ so that the columns have unit length (using a diagonal matrix L). Then we factor C_n using SVD to form

$$C_n = W \Sigma V^T, \tag{15}$$

where we set all but the first 4 singular values to zero. Equivalently, we have W is $3J \times 4$, Σ is 4×4, and V^T is $4 \times N$.

We can rewrite the n^{th} column of C_n as $Z_n \vec{z}_n$, where Z_n is a $3J \times N$ matrix obtained from the image points $\vec{p}_{j,n}$ and the n^{th} weight $L_{n,n}$. Here $\vec{z}_n = (z_{1,n}, \ldots, z_{J,n})^T$, which are the projective depths for the n^{th} point in each of the J frames. We then update $\vec{z}_n$ to better match the current factorization. That is, we wish to minimize

$$||Z_n \vec{z}_n - W \Sigma V^T \vec{e}_n|| \tag{16}$$

wrt $\vec{z}_n$, subject to the constraint that the updated column of C_n still has unit length, i.e., $||Z_n \vec{z}_n|| = 1$. Here $\vec{e}_n$ is the n^{th} standard unit vector, $e_{n,i} = \delta_{n,i}$. (In `projectiveMassageDino.m` this update of $\vec{z}_n$ is done with one step along the gradient direction for this constrained optimization problem.) Once all the projective depths have been updated, we reform the normalized data matrix C_n, and redo the factorization (15). This process is iterated until convergence.

Projective Reconstruction

Upon convergence we have a projective factorization $C_n = W\Sigma V^T$. As in the orthographic case, this factorization is only unique up to a nonsingular matrix H. In this case, H is a 4×4, 3D homography matrix. In particular, we have the factorization, $C = C_n L^{-1} = MP$ with

$$\begin{aligned} P &= H^{-1}\Sigma V^T L^{-1} \\ M &= WH. \end{aligned} \tag{17}$$

Since the shape matrix P is known up to a 3D homography H, this is called a **projective reconstruction**.

This projective reconstruction can be "upgraded" to a metric reconstruction by using information about the camera matrices M_j to constrain the 3D homography matrix H. In order to understand this, we must first introduce the absolute dual quadric from projective geometry.

Absolute Dual Quadric (Canonical Coords)

The equation of a plane in 3D is

$$\vec{m}^{\,T}\vec{P} = 0,$$

where $\vec{P} = (X, Y, Z, 1)^T$ is a 3D point written in homogeneous coordinates.

Imagine two planes, with coefficient vectors $\vec{m}_1$ and $\vec{m}_2$. Then the angle between these two planes is θ where

$$\cos(\theta) = \frac{\vec{m}_1^T \hat{Q}_\infty \vec{m}_2}{\sqrt{(\vec{m}_1^T \hat{Q}_\infty \vec{m}_1)(\vec{m}_2^T \hat{Q}_\infty \vec{m}_2)}}. \tag{18}$$

Here $\hat{Q}_\infty$ is the absolute dual quadric in canonical coordinates,

$$\hat{Q}_\infty = \begin{pmatrix} I_3 & \vec{0} \\ \vec{0}^{\,T} & 0 \end{pmatrix}. \tag{19}$$

Absolute Dual Quadric (General Coords)

Suppose H is any nonsingular 3D homography matrix, and consider the projective coordinates $\vec{P}' = H\vec{P}$. The planes $\vec{m}_k \cdot \vec{P} = 0$ can be expressed in these new coordinates as

$$\vec{m}'_k \cdot \vec{P}' = 0, \quad \text{where } \vec{m}'_k = H^{-T}\vec{m}_k.$$

We can measure the same angle between these two planes using the absolute dual quadric in general projective coordinates, namely

$$Q_\infty = H\hat{Q}_\infty H^T. \tag{20}$$

In fact, it follows that

$$\cos(\theta) = \frac{(\vec{m}'_1)^T Q_\infty \vec{m}'_2}{\sqrt{((\vec{m}'_1)^T Q_\infty \vec{m}'_1)((\vec{m}'_2)^T Q_\infty \vec{m}'_2)}}.$$

The general idea behind upgrading a projective reconstruction to a metric one is to use the absolute dual quadric to express known properties of the camera coordinates, such as the fact that the planes perpendicular to the X, Y, and Z axes are mutually perpendicular (i.e. $\cos(\theta) = 0$).

Upgrading to a Metric Reconstruction

In particular, from (2) and (3) it follows that

$$M_j \hat{Q}_\infty M_j^T = \begin{pmatrix} f_j^2 & 0 & 0 \\ 0 & f_j^2 & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{21}$$

This is the analogue of equation (11) in the orthographic case. From equation (17) we also have $M = WH$, where W is known from the projective factorization. So

$$M_j \hat{Q}_\infty M_j^T = W_j H \hat{Q}_\infty H^T W_j^T = W_j Q_\infty W_j^T. \tag{22}$$

We see that, for the j^{th} camera, (22) and (21) provide 5 linear equations for Q_∞ (6 linear equations if f_j is known). So we have (at least) $5J$ linear constraints on Q_∞.

Since we know Q_∞ is a symmetric 4×4 matrix there are only 10 degrees of freedom to determine, and $J = 2$ frames are enough. (Note we also know Q_∞ has rank 3 and has non-negative eigenvalues.)

Solving for H

Given Q_∞ (which is symmetric, positive semi-definite, rank 3) we can compute its eigenvalue decompositon

$$Q_\infty = U_q \Lambda_q U_q^T, \quad \Lambda_q = \mathrm{diag}[\lambda_1, \lambda_2, \lambda_3, 0],$$

with $\lambda_i > 0$. It then follows from (20) that

$$H = U_q \mathrm{diag}[\lambda_1^{1/2}, \lambda_2^{1/2}, \lambda_3^{1/2}, 1] A, \tag{23}$$

where A is the matrix for a general 3D similarity transform

$$A = \begin{pmatrix} R & \vec{d} \\ \vec{0}^T & s \end{pmatrix}.$$

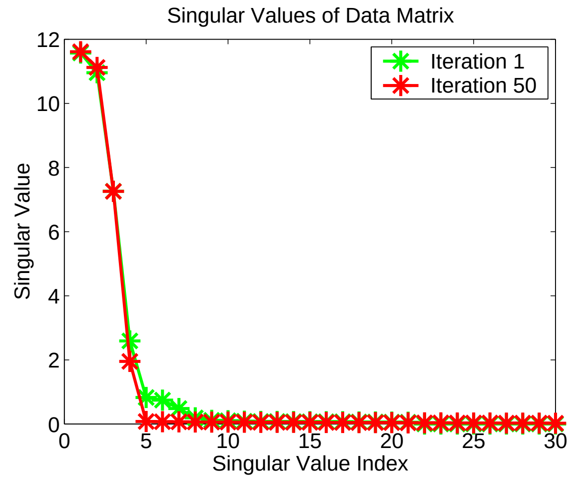
Here R is a unitary matrix. The reason A remains unknown is that the absolute dual quadric $\hat{Q}_\infty$ is invariant to similarity transformations

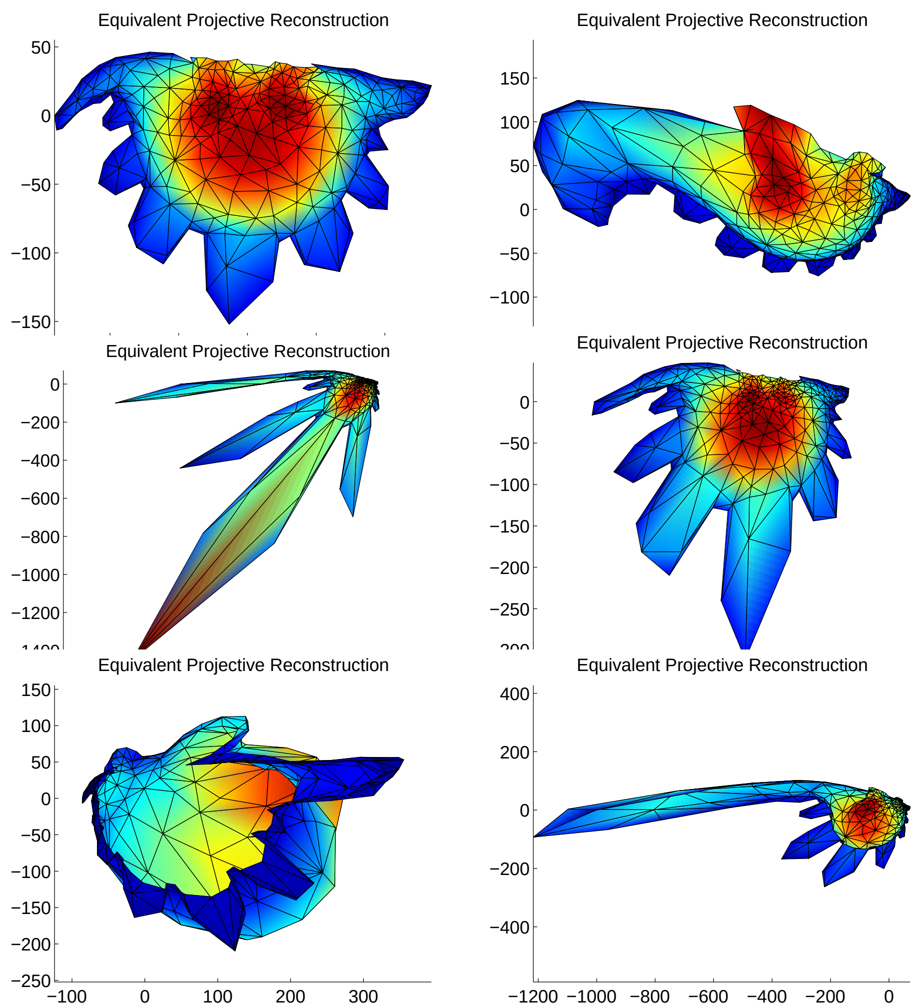
$$A \hat{Q}_\infty A^T = \hat{Q}_\infty.$$

This is easy to verify from the forms of A and $\hat{Q}_\infty$.

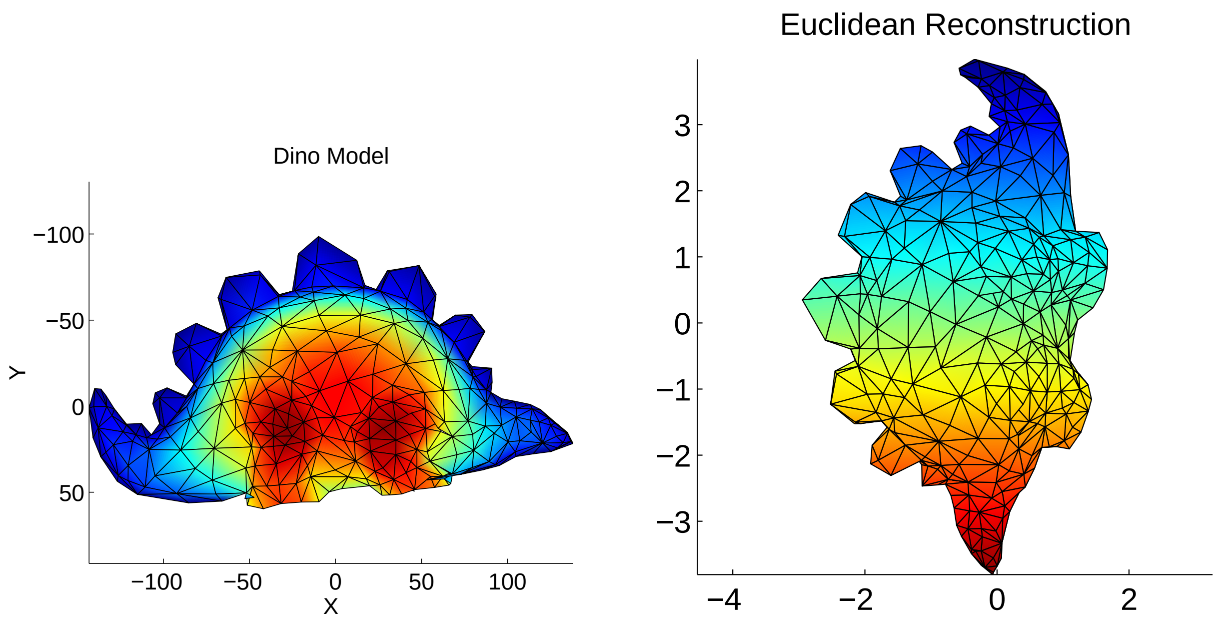
Finally, equations (2, 3, 17) can be used to remove the reflection ambiguity. The only remaining ambiguities are the overall orientation, origin and scale of the world coordinate frame.

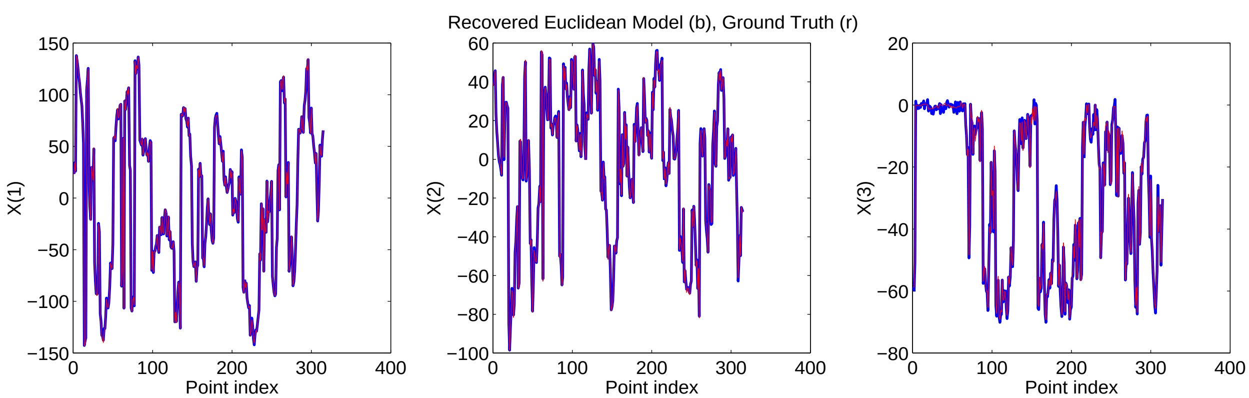
Dino Example, Projective Case





Dino Example, Projective Case





Structure from Motion

The use of the theoretical rank for a set of observations provides a key insight into the structure from motion problems (see Jepson and Heeger, 1991).

Consider a camera travelling through a stationary environment. Then the scene appears to move with translational velocity $\vec{T}$ and angular velocity $\vec{\Omega}$. In the camera's coordinates, the motion of any scene point $\vec{X}$ is

$$\frac{d\vec{X}}{dt} = \vec{T} + \vec{\Omega} \times \vec{X}.$$

Suppose we observe the motion field $\vec{u}(\vec{x}_k)$ at K image points, $\{\vec{x}_k\}_{k=1}^{K}$, in this camera's image. Let $\vec{X}(\vec{x}_k)$ be the 3D scene point associated with the k^{th} image point $\vec{x}_k$. Then it can be shown that $\vec{U}^{\,T} \equiv (\vec{u}_1^{\,T}, \ldots, \vec{u}_K^{\,T})$ satisfies

$$\vec{U} = C(\vec{T}) \begin{pmatrix} \vec{z} \\ \vec{\Omega} \end{pmatrix} = A(\vec{T})\vec{z} + B\vec{\Omega}. \tag{24}$$

Here $\vec{z}$ is a K-vector, with elements $z_k = 1/||\vec{X}(\vec{x}_k)||$, $A(\vec{T})$ is a $2K \times K$ matrix that depends linearly on $\vec{T}$, and B is a $2K \times 3$ matrix that depends only on the image points $\vec{x}_k$.

Notice, for $\vec{T} = \vec{0}$ we have $A(\vec{T}) = 0$, and (24) states that the flow field $\vec{U}$ must be in the rank 3 subspace formed by the range of the matrix B. Similarly, for nonzero $\vec{T}$, equation (24) states that the $2K$-dimensional flow field $\vec{U}$ must be in the $K+3$-dimensional subspace formed by the range of $C(\vec{T})$.

This range condition can be used to identify $\vec{T}$ (up to a speed ambiguity, i.e., $||\vec{T}||$ remains unknown) and $\vec{\Omega}$ given the motion field $\vec{U}$. Moreover, given $\vec{T}/||\vec{T}||$ and $\vec{\Omega}$, equation (24) can be used to solve for the inverse depths $\vec{z}$ (up to an overall scale ambiguity).

Further Readings

Hartley and Zisserman, Multiple View Geometry in Computer Vision, Cambridge Univ. Press, 2000.

Jepson and Heeger, A fast subspace algorithm for recovering rigid motion, Proc. IEEE Workshop on Visual Motion, Princeton, 1991, pp. 124-131.

Koenderink and van Doorn, Affine structure from motion, Journal of the Optical Society of America, 8(2), 1991, pp. 377-385.

S. Mahamud, M. Hebert, Y. Omori and J. Ponce, Provably-Convergent Iterative Methods for Projective Structure from Motion, CVPR 2001.

Polleyfeys, Koch and van Gool, Self-calibration and metric reconstruction in spite of varying and unknown intrinsic camera parameters, IJCV 1999.

Tomasi and Kanade, Shape and motion from image streams under orthography: A factorization approach, IJCV, Vol. 9 (No. 2), 1992, pp.137-154.