

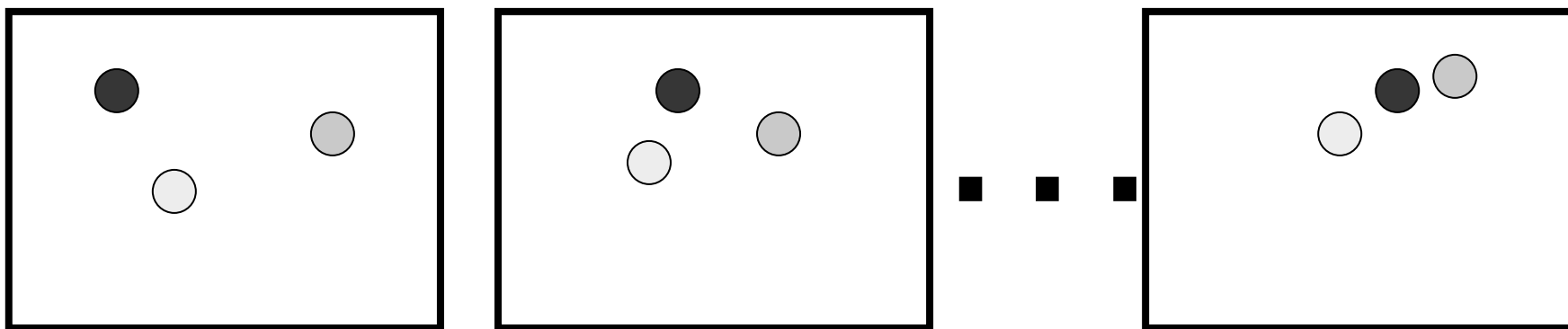
Structure-from-Motion

- Determining the 3-D structure of the world, and/or the motion of a camera using a sequence of images taken by a moving camera.
 - Equivalently, we can think of the world as moving and the camera as fixed.
- Like stereo, but the position of the camera isn't known (and it's more natural to use many images with little motion between them, not just two with a lot of motion).
 - We may or may not assume we know the parameters of the camera, such as its focal length.

Structure-from-Motion

- As with stereo, we can divide problem:
 - Correspondence.
 - Reconstruction.
- Again, we'll talk about reconstruction first.
 - So for the next few classes we assume that each image contains some points, and we know which points match which.

Structure-from-Motion



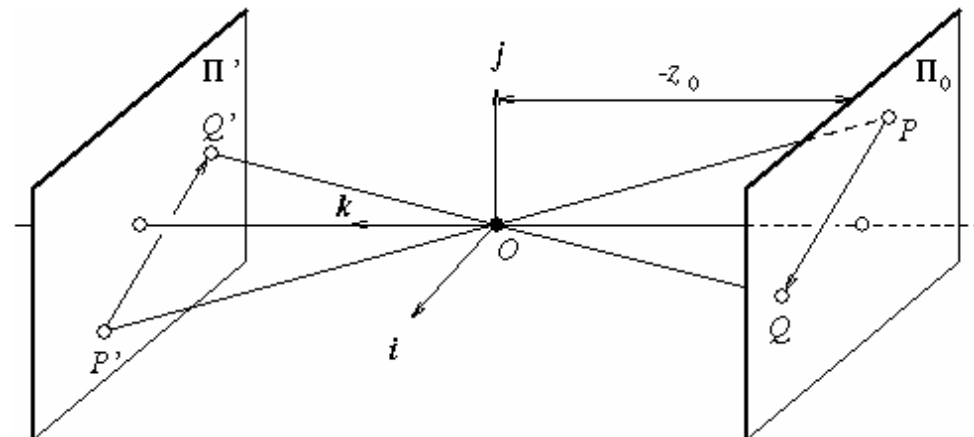
Movie

Reconstruction

- A lot harder than with stereo.
- Start with simpler case: scaled orthographic projection (weak perspective).
 - Recall, in this we remove the z coordinate and scale all x and y coordinates the same amount.

Weak perspective (scaled orthographic projection)

- Issue
 - perspective effects, but not over the scale of individual objects
 - collect points into a group at about the same depth, then divide each point by the depth of its group



(Forsyth & Ponce)

Perspective \rightarrow Scaled Orthographic

- Recall: $(x_i, y_i, z_i) \rightarrow (x_i/z_i, y_i/z_i)$
- Let $Z = (z_1 + z_2 + \dots + z_n)/n$
- Then, $(x_i, y_i, z_i) \text{ approx-} \rightarrow (x_i/Z, y_i/Z)$

The Equation of Weak Perspective

$$(x, y, z) \rightarrow s(x, y) \quad \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} s & 0 & 0 & 0 \\ 0 & s & 0 & 0 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

- s is constant for all points.
- Parallel lines no longer converge, they remain parallel.

Pros and Cons of These Models

- Weak perspective much simpler math.
 - Accurate when object is small and distant.
 - Most useful for recognition.
- Pinhole perspective much more accurate for scenes.
 - Used in structure from motion.
- When accuracy really matters, must model real cameras.

First: Represent motion

- We'll talk about a fixed camera, and moving object.
- Key point:

Points

$$P = \begin{pmatrix} x_1 & x_2 & \cdot & \cdot & \cdot & x_n \\ y_1 & y_2 & \cdot & \cdot & \cdot & y_n \\ z_1 & z_2 & \cdot & \cdot & \cdot & z_n \\ 1 & 1 & \cdot & \cdot & \cdot & 1 \end{pmatrix}$$

Some matrix

$$S = \begin{pmatrix} s_{1,1} & s_{1,2} & s_{1,3} & t_x \\ s_{2,1} & s_{2,2} & s_{2,3} & t_y \end{pmatrix}$$

The image

$$I = \begin{pmatrix} u_1 & u_2 & \cdot & \cdot & \cdot & u_n \\ v_1 & v_2 & \cdot & \cdot & \cdot & v_n \end{pmatrix}$$

Then: $I = SP$

Remember what this means.

- We are representing moving a set of points, projecting them into the image, and scaling them.
- Matrix multiplication: take inner product between each row of S and each point. First row of S produces X coordinates, while second row produces Y .
- Projection occurs because S has no third row.
- Translation occurs with t_x and t_y .
- Scaling can be encoded with a scale factor in S .
- The rest of S must be allowing the object to rotate.

Examples:

- $S = [s, 0, 0, 0; 0, s, 0, 0]$; This is just projection, with scaling by s .
- $S = [s, 0, 0, s*tx; 0, s, 0, s*ty]$; This is translation by $(tx, ty, something)$, projection, and scaling.

Structure-from-Motion

- S encodes:
 - Projection: only two lines
 - Scaling, since S can have a scale factor.
 - Translation, by t_x/s and t_y/s .
 - Rotation:

$$I = SP$$

Rotation

$$\begin{pmatrix} r_{1,1} & r_{1,2} & r_{1,3} \\ r_{2,1} & r_{2,2} & r_{2,3} \\ r_{3,1} & r_{3,2} & r_{3,3} \end{pmatrix} P$$

Represents a
3D rotation of
the points in P .

First, look at 2D rotation (easier)

Matrix R acts
on points by
rotating them.

$$R = \begin{pmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{pmatrix}$$

$$\begin{pmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{pmatrix} \begin{pmatrix} x_1 & x_2 & \cdot & \cdot & \cdot & x_n \\ y_1 & y_2 & & & & y_n \end{pmatrix}$$

- Also, $RR^T = \text{Identity}$. R^T is also a rotation matrix, in the opposite direction to R .

Why does multiplying points by R rotate them?

- Think of the rows of R as a new coordinate system.

Taking inner products of each point with these expresses that point in that coordinate system.

- This means rows of R must be orthonormal vectors (orthogonal unit vectors).
- Think of what happens to the points $(1,0)$ and $(0,1)$. They go to $(\cos \theta, -\sin \theta)$, and $(\sin \theta, \cos \theta)$. They remain orthonormal, and rotate clockwise by θ .
 - Any other point, (a,b) can be thought of as $a(1,0) + b(0,1)$. $R(a(1,0)+b(0,1)) = Ra(1,0) + Rb(0,1) = aR(1,0) + bR(0,1)$. So it's in the same position relative to the rotated coordinates that it was in before rotation relative to the x, y coordinates. That is, it's rotated.

Simple 3D Rotation

$$\begin{pmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 & x_2 & \cdot & \cdot & \cdot & x_n \\ y_1 & y_2 & & & & y_n \\ z_1 & z_2 & & & & z_n \end{pmatrix}$$

Rotation about z axis.

Rotates x,y coordinates. Leaves z coordinates fixed.

Full 3D Rotation

$$R = \begin{pmatrix} \cos \theta & \sin \theta & 0 \\ -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos \beta & 0 & \sin \beta \\ 0 & 1 & 0 \\ -\sin \beta & 0 & \cos \beta \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos \alpha & \sin \alpha \\ 0 & -\sin \alpha & \cos \alpha \end{pmatrix}$$

- Any rotation can be expressed as combination of three rotations about three axes.

$$RR^T = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

- Rows (and columns) of R are orthonormal vectors.
- R has determinant 1 (not -1).

- Intuitively, it makes sense that 3D rotations can be expressed as 3 separate rotations about fixed axes. Rotations have 3 degrees of freedom; two describe an axis of rotation, and one the amount.
- Rotations preserve the length of a vector, and the angle between two vectors. Therefore, $(1,0,0)$, $(0,1,0)$, $(0,0,1)$ must be orthonormal after rotation. After rotation, they are the three columns of R . So these columns must be orthonormal vectors for R to be a rotation. Similarly, if they are orthonormal vectors (with determinant 1) R will have the effect of rotating $(1,0,0)$, $(0,1,0)$, $(0,0,1)$. Same reasoning as 2D tells us all other points rotate too.
 - Note if R has determinant -1 , then R is a rotation plus a reflection.

Questions?

Putting it Together

$$\begin{array}{c}
 \text{Scale} \\
 \swarrow \\
 S
 \end{array}
 \begin{array}{c}
 \text{Projection} \\
 \left(\begin{array}{ccc} 1 & 0 & 0 \\ 0 & 1 & 0 \end{array} \right)
 \end{array}
 \begin{array}{c}
 \text{3D Translation} \\
 \left(\begin{array}{ccc|c} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \end{array} \right)
 \end{array}
 \begin{array}{c}
 \text{3D Rotation} \\
 \left(\begin{array}{ccc|c} r_{1,1} & r_{1,2} & r_{1,3} & 0 \\ r_{2,1} & r_{2,2} & r_{2,3} & 0 \\ r_{3,1} & r_{3,2} & r_{3,3} & 0 \\ 0 & 0 & 0 & 1 \end{array} \right)
 \end{array}
 P$$

$$\equiv \left(\begin{array}{ccc|c} s_{1,1} & s_{1,2} & s_{1,3} & st_x \\ s_{2,1} & s_{2,2} & s_{2,3} & st_y \end{array} \right) P$$

where

$$(s_{1,1}, s_{1,2}, s_{1,3}) \bullet (s_{2,1}, s_{2,2}, s_{2,3}) = 0$$

$$\|(s_{1,1}, s_{1,2}, s_{1,3})\| = \|(s_{2,1}, s_{2,2}, s_{2,3})\|$$

We can just write st_x as t_x and st_y as t_y .

Affine Structure from Motion


$$\begin{pmatrix} s_{1,1} & s_{1,2} & s_{1,3} & t_x \\ s_{2,1} & s_{2,2} & s_{2,3} & t_y \end{pmatrix} P$$

~~where~~

$$\langle (s_{1,1}, s_{1,2}, s_{1,3}) \bullet (s_{2,1}, s_{2,2}, s_{2,3}) \rangle = 0$$

$$\| (s_{1,1}, s_{1,2}, s_{1,3}) \| = \| (s_{2,1}, s_{2,2}, s_{2,3}) \|$$

Affine Structure-from-Motion: Two Frames (1)

$$\begin{pmatrix} u_1^1 & u_2^1 & \cdot & \cdot & \cdot & u_n^1 \\ v_1^1 & v_2^1 & & & & v_n^1 \end{pmatrix} \begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 & t_x^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 & t_y^1 \end{pmatrix} \begin{pmatrix} x_1 & x_2 & \cdot & \cdot & \cdot & x_n \\ y_1 & y_2 & & & & y_n \\ z_1 & z_2 & & & & z_n \\ 1 & 1 & & & & 1 \end{pmatrix}$$


Affine Structure-from-Motion: Two Frames (2)

To make things
easy, suppose:

$$\begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \\ z_1 & z_2 & z_3 & z_4 \\ 1 & 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

Affine Structure-from-Motion: Two Frames (3)

$$\begin{pmatrix} u_1^1 & u_2^1 & \cdot & \cdot & \cdot & u_n^1 \\ v_1^1 & v_2^1 & & & & v_n^1 \\ u_1^2 & u_2^2 & & & & u_n^2 \\ v_1^2 & v_2^2 & & & & v_n^2 \end{pmatrix} = \begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 & t_x^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 & t_y^1 \\ s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 & t_x^2 \\ s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 & t_y^2 \end{pmatrix} \begin{pmatrix} x_1 & x_2 & \cdot & \cdot & \cdot & x_n \\ y_1 & y_2 & & & & y_n \\ z_1 & z_2 & & & & z_n \\ 1 & 1 & & & & 1 \end{pmatrix}$$

Looking at the first four points, we get:

$$\begin{pmatrix} u_1^1 & u_2^1 & u_3^1 & u_4^1 \\ v_1^1 & v_2^1 & v_3^1 & v_4^1 \\ u_1^2 & u_2^2 & u_3^2 & u_4^2 \\ v_1^2 & v_2^2 & v_3^2 & v_4^2 \end{pmatrix} = \begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 & t_x^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 & t_y^1 \\ s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 & t_x^2 \\ s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 & t_y^2 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

Affine Structure-from-Motion: Two Frames (4)

$$\begin{pmatrix} u_1^1 & u_2^1 & u_3^1 & u_4^1 \\ v_1^1 & v_2^1 & v_3^1 & v_4^1 \\ u_1^2 & u_2^2 & u_3^2 & u_4^2 \\ v_1^2 & v_2^2 & v_3^2 & v_4^2 \end{pmatrix} = \begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 & t_x^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 & t_y^1 \\ s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 & t_x^2 \\ s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 & t_y^2 \end{pmatrix} \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

We can solve for motion by inverting matrix of points.

Or, explicitly, we see that first column on left (images of first point) give the translations. After solving for these, we can solve for the each column of the s components of the motion using the images of each point, in turn.

Affine Structure-from-Motion: Two Frames (5)

$$\begin{pmatrix} u_k^1 \\ v_k^1 \\ u_k^2 \\ v_k^2 \end{pmatrix} = \begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 & t_x^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 & t_y^1 \\ s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 & t_x^2 \\ s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 & t_y^2 \end{pmatrix} \begin{pmatrix} x_k \\ y_k \\ z_k \\ 1 \end{pmatrix}$$

Once we know the motion, we can use the images of another point to solve for the structure. We have four linear equations, with three unknowns.

Affine Structure-from-Motion: Two Frames (6)

Suppose we just know where the k 'th point is in image 1.

$$\begin{pmatrix} u_k^1 \\ v_k^1 \\ ? \\ ? \end{pmatrix} = \begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 & t_x^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 & t_y^1 \\ s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 & t_x^2 \\ s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 & t_y^2 \end{pmatrix} \begin{pmatrix} x_k \\ y_k \\ z_k \\ 1 \end{pmatrix}$$

Then, we can use the first two equations to write x_k and y_k as linear in z_k . The final two equations lead to two linear equations in the missing values and z_k . If we eliminate z_k we get one linear equation in the missing values. This means the unknown point lies on a known line. That is, we recover the epipolar constraint. Furthermore, these lines are all parallel.

Affine Structure-from-Motion: Two Frames (7)

But, what if the first four points aren't so simple?

Then we define A so that:

$$A \begin{pmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \\ z_1 & z_2 & z_3 & z_4 \\ 1 & 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

This is always possible as long as the points aren't coplanar.

Affine Structure-from-Motion: Two Frames (8)

Then,
given:

$$\begin{pmatrix} u_1^1 & u_2^1 & \cdot & \cdot & \cdot & u_n^1 \\ v_1^1 & v_2^1 & & & & v_n^1 \\ u_1^2 & u_2^2 & & & & u_n^2 \\ v_1^2 & v_2^2 & & & & v_n^2 \end{pmatrix} = \begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 & t_x^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 & t_y^1 \\ s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 & t_x^2 \\ s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 & t_y^2 \end{pmatrix} \begin{pmatrix} x_1 & x_2 & \cdot & \cdot & \cdot & x_n \\ y_1 & y_2 & & & & y_n \\ z_1 & z_2 & & & & z_n \\ 1 & 1 & & & & 1 \end{pmatrix}$$

We have:

$$\begin{pmatrix} u_1^1 & u_2^1 & \cdot & \cdot & \cdot & u_n^1 \\ v_1^1 & v_2^1 & & & & v_n^1 \\ u_1^2 & u_2^2 & & & & u_n^2 \\ v_1^2 & v_2^2 & & & & v_n^2 \end{pmatrix} = \begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 & t_x^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 & t_y^1 \\ s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 & t_x^2 \\ s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 & t_y^2 \end{pmatrix} A^{-1} A \begin{pmatrix} x_1 & x_2 & \cdot & \cdot & \cdot & x_n \\ y_1 & y_2 & & & & y_n \\ z_1 & z_2 & & & & z_n \\ 1 & 1 & & & & 1 \end{pmatrix}$$

And:

$$\begin{pmatrix} u_1^1 & u_2^1 & \cdot & \cdot & \cdot & u_n^1 \\ v_1^1 & v_2^1 & & & & v_n^1 \\ u_1^2 & u_2^2 & & & & u_n^2 \\ v_1^2 & v_2^2 & & & & v_n^2 \end{pmatrix} = \begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 & t_x^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 & t_y^1 \\ s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 & t_x^2 \\ s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 & t_y^2 \end{pmatrix} A^{-1} \begin{pmatrix} 0 & 1 & 0 & 0 & \cdot & \cdot & \cdot & x_n \\ 0 & 0 & 1 & 0 & & & & y_n \\ 0 & 0 & 0 & 1 & & & & z_n \\ 1 & 1 & 1 & 1 & & & & 1 \end{pmatrix}$$

Affine Structure-from-Motion: Two Frames (9)

Given:

$$\begin{pmatrix} u_1^1 & u_2^1 & \cdot & \cdot & \cdot & u_n^1 \\ v_1^1 & v_2^1 & & & & v_n^1 \\ u_1^2 & u_2^2 & & & & u_n^2 \\ v_1^2 & v_2^2 & & & & v_n^2 \end{pmatrix} = \begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 & t_x^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 & t_y^1 \\ s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 & t_x^2 \\ s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 & t_y^2 \end{pmatrix} A^{-1} \begin{pmatrix} 0 & 1 & 0 & 0 & \cdot & \cdot & \cdot & x_n \\ 0 & 0 & 1 & 0 & & & & y_n \\ 0 & 0 & 0 & 1 & & & & z_n \\ 1 & 1 & 1 & 1 & & & & 1 \end{pmatrix}$$

Then we just pretend that:

$$\begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 & t_x^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 & t_y^1 \\ s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 & t_x^2 \\ s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 & t_y^2 \end{pmatrix} A^{-1}$$

is our motion,
and solve as
before.

Affine Structure-from-Motion: Two Frames (10)

This means that we can never determine the exact 3D structure of the scene. We can only determine it up to some transformation, A . Since if a structure and motion explains the points:

$$\begin{pmatrix} u_1^1 & u_2^1 & \cdot & \cdot & \cdot & u_n^1 \\ v_1^1 & v_2^1 & & & & v_n^1 \\ u_1^2 & u_2^2 & & & & u_n^2 \\ v_1^2 & v_2^2 & & & & v_n^2 \end{pmatrix} = \begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 & t_x^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 & t_y^1 \\ s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 & t_x^2 \\ s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 & t_y^2 \end{pmatrix} \begin{pmatrix} x_1 & x_2 & \cdot & \cdot & \cdot & x_n \\ y_1 & y_2 & & & & y_n \\ z_1 & z_2 & & & & z_n \\ 1 & 1 & & & & 1 \end{pmatrix}$$

So does
another of
the form:

$$\begin{pmatrix} u_1^1 & u_2^1 & \cdot & \cdot & \cdot & u_n^1 \\ v_1^1 & v_2^1 & & & & v_n^1 \\ u_1^2 & u_2^2 & & & & u_n^2 \\ v_1^2 & v_2^2 & & & & v_n^2 \end{pmatrix} = \left(\begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 & t_x^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 & t_y^1 \\ s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 & t_x^2 \\ s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 & t_y^2 \end{pmatrix} A^{-1} \right) \left(A \begin{pmatrix} x_1 & x_2 & \cdot & \cdot & \cdot & x_n \\ y_1 & y_2 & & & & y_n \\ z_1 & z_2 & & & & z_n \\ 1 & 1 & & & & 1 \end{pmatrix} \right)$$

Affine Structure-from-Motion: Two Frames (11)

$$\begin{pmatrix} u_1^1 & u_2^1 & \cdot & \cdot & \cdot & u_n^1 \\ v_1^1 & v_2^1 & & & & v_n^1 \\ u_1^2 & u_2^2 & & & & u_n^2 \\ v_1^2 & v_2^2 & & & & v_n^2 \end{pmatrix} = \left(\begin{pmatrix} s_{1,1}^1 & s_{1,2}^1 & s_{1,3}^1 & t_x^1 \\ s_{2,1}^1 & s_{2,2}^1 & s_{2,3}^1 & t_y^1 \\ s_{1,1}^2 & s_{1,2}^2 & s_{1,3}^2 & t_x^2 \\ s_{2,1}^2 & s_{2,2}^2 & s_{2,3}^2 & t_y^2 \end{pmatrix} A^{-1} \right) \left(A \begin{pmatrix} x_1 & x_2 & \cdot & \cdot & \cdot & x_n \\ y_1 & y_2 & & & & y_n \\ z_1 & z_2 & & & & z_n \\ 1 & 1 & & & & 1 \end{pmatrix} \right)$$

Note that A has
the form:

A corresponds to
translation of the
points, plus a
linear
transformation.

$$A = \begin{pmatrix} a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \\ a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

For example, there is clearly a translational ambiguity in recovering the points. We can't tell the difference between two point sets that are identical up to a translation when we only see them after they undergo an unknown translation. Similarly, there's clearly a rotational ambiguity. The rest of the ambiguity is a stretching in an unknown direction.

Let's take an explicit example of this. Suppose we have a cube with vertices at: (0,0,0) (10,0,0), (0,0,10)..... Suppose we transform this by rotating it by 45 degrees about the y axis. Then the transformation matrix is: $[\sqrt{2} \ 0 \ -\sqrt{2} \ 0; 0 \ 1 \ 0 \ 0]$. Now suppose instead we had a rectanguloid with corners at (10, 0, 0) (11,0,0), (10, 0, 10) We can transform this rectanguloid into the first cube by transforming it as: $[10 \ 0 \ 0 \ -100; 0 \ 1 \ 0 \ 0; 0 \ 0 \ 1 \ 0; 0 \ 0 \ 0 \ 1]$. Then we can apply $[\sqrt{2} \ 0 \ -\sqrt{2} \ 0; 0 \ 1 \ 0 \ 0]$ to the resulting cube, to generate the same image. Or, we could have combined these two transformations into one:

$$[\sqrt{2} \ 0 \ -\sqrt{2} \ 0; 0 \ 1 \ 0 \ 0] * [10 \ 0 \ 0 \ -100; 0 \ 1 \ 0 \ 0; 0 \ 0 \ 1 \ 0; 0 \ 0 \ 0 \ 1]$$

$$= [10\sqrt{2} \ 0 \ -\sqrt{2} \ 0; 0 \ 1 \ 0 \ 0]$$