Structure from Motion

CS4670/CS5670 - Kevin Matzen - April 15, 2016

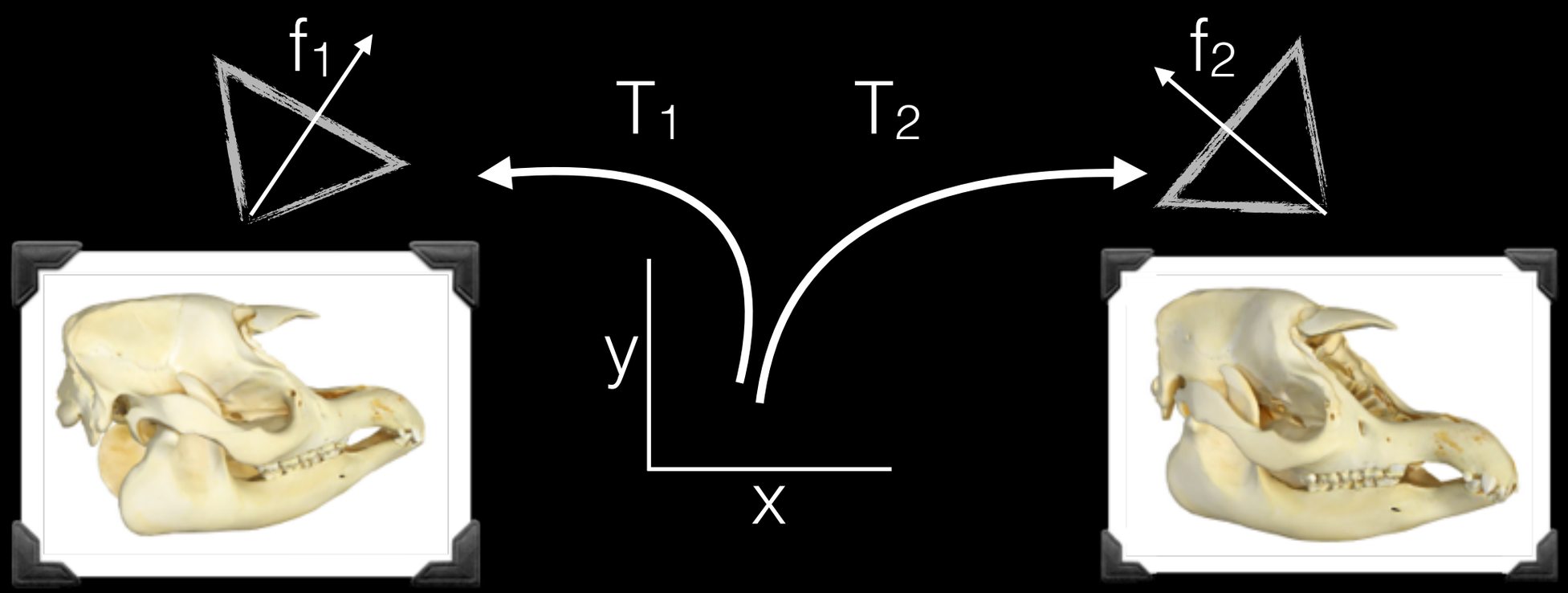
Video credit: Agarwal, et. al. Building Rome in a Day, ICCV 2009

Roadmap

- What we've seen so far
  - Single view modeling (1 camera)
  - Stereo modeling (2 cameras)
  - Multi-view stereo (3+ cameras)
- How do we recover camera parameters necessary for MVS?

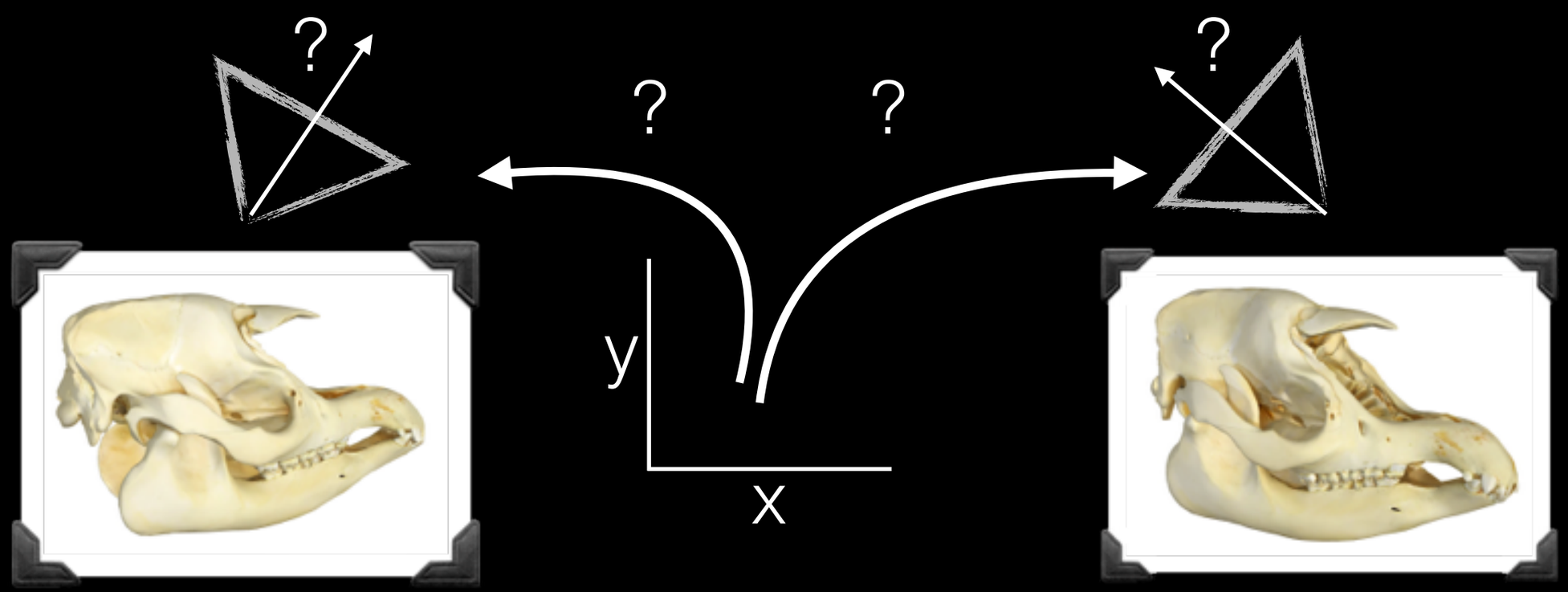
Wednesday's Lecture

Assume we are always given the camera calibration.

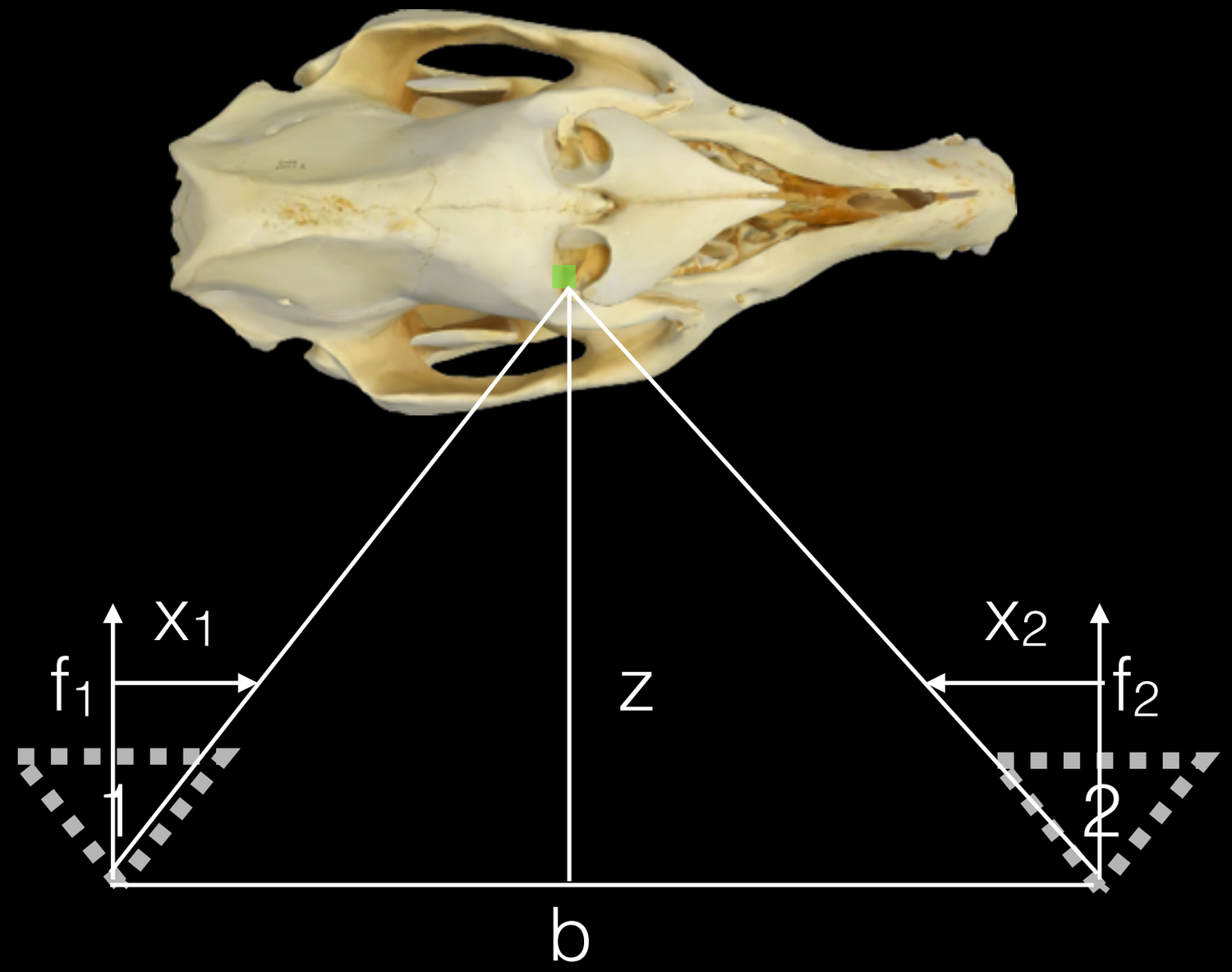


Today's Lecture

Assume we are ~~always~~ **never** given the camera calibration.



Calibration makes 3D reasoning possible!



Today's outline

- How can we calibrate our cameras?
- How can we calibrate a camera without photos of a calibration target?
- How can we automate this calibration at scale?

Projection Model

$$\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \cong \begin{bmatrix} m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{11} & m_{12} & m_{13} \\ m_{20} & m_{21} & m_{22} & m_{23} \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$

Projection Model

$$\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \cong \begin{bmatrix} m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{11} & m_{12} & m_{13} \\ m_{20} & m_{21} & m_{22} & m_{23} \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$

Some 3D world-space point

Projection Model

$$\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \cong \begin{bmatrix} m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{11} & m_{12} & m_{13} \\ m_{20} & m_{21} & m_{22} & m_{23} \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$

A 2D image-space projection

Some 3D world-space point

Projection Model

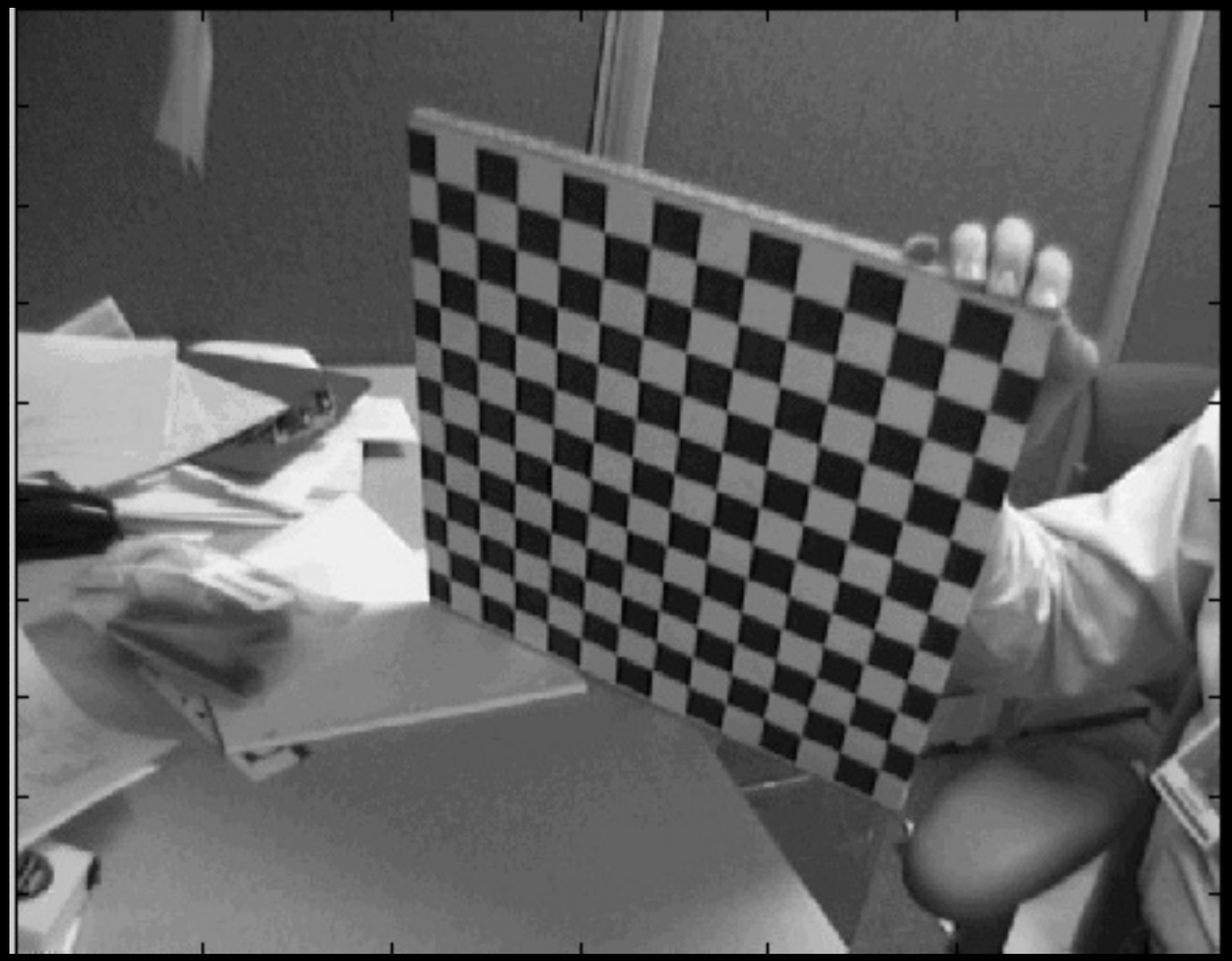
Calibration gives us these

$$\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \cong \begin{bmatrix} m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{11} & m_{12} & m_{13} \\ m_{20} & m_{21} & m_{22} & m_{23} \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$

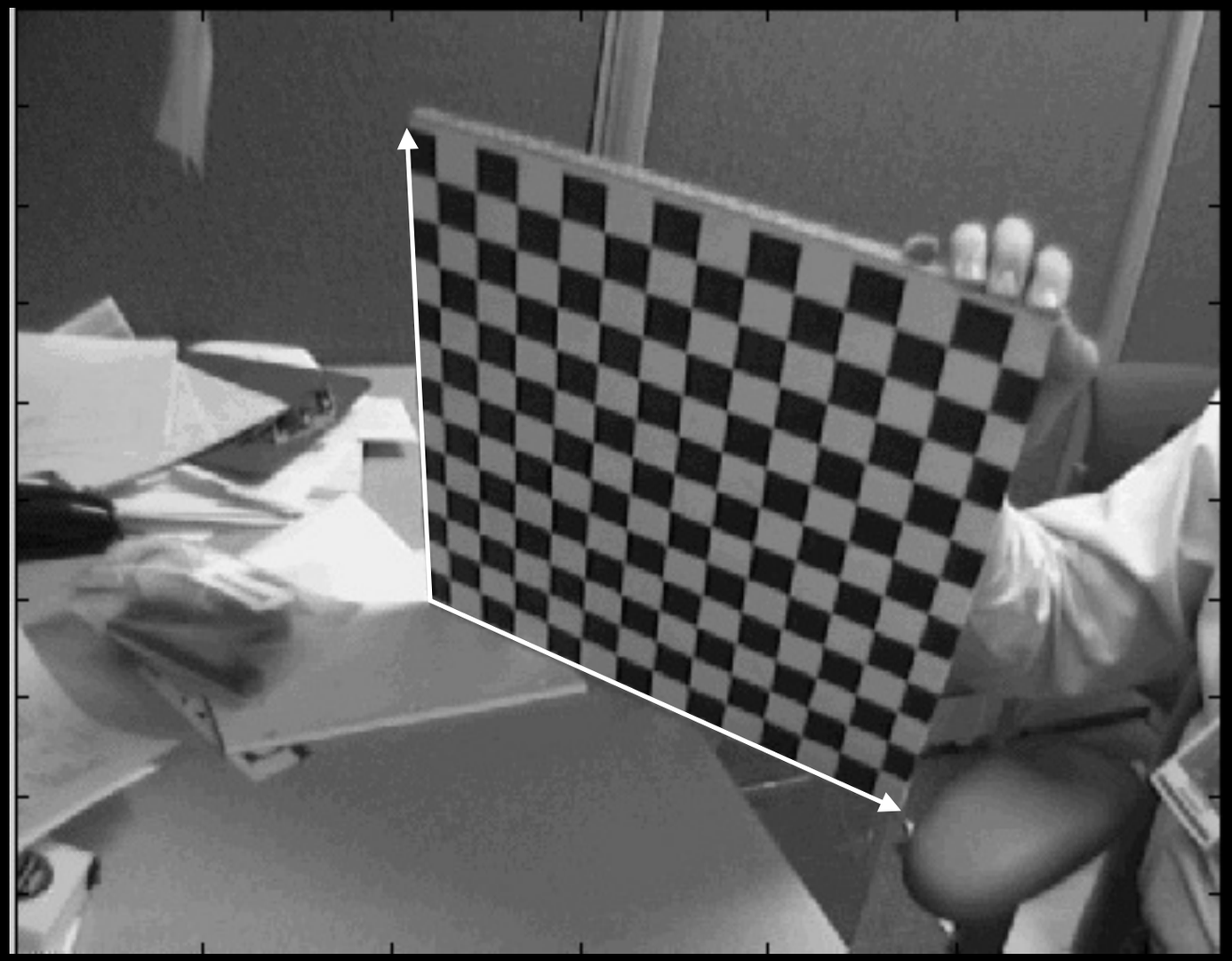
A 2D image-space projection

Some 3D world-space point

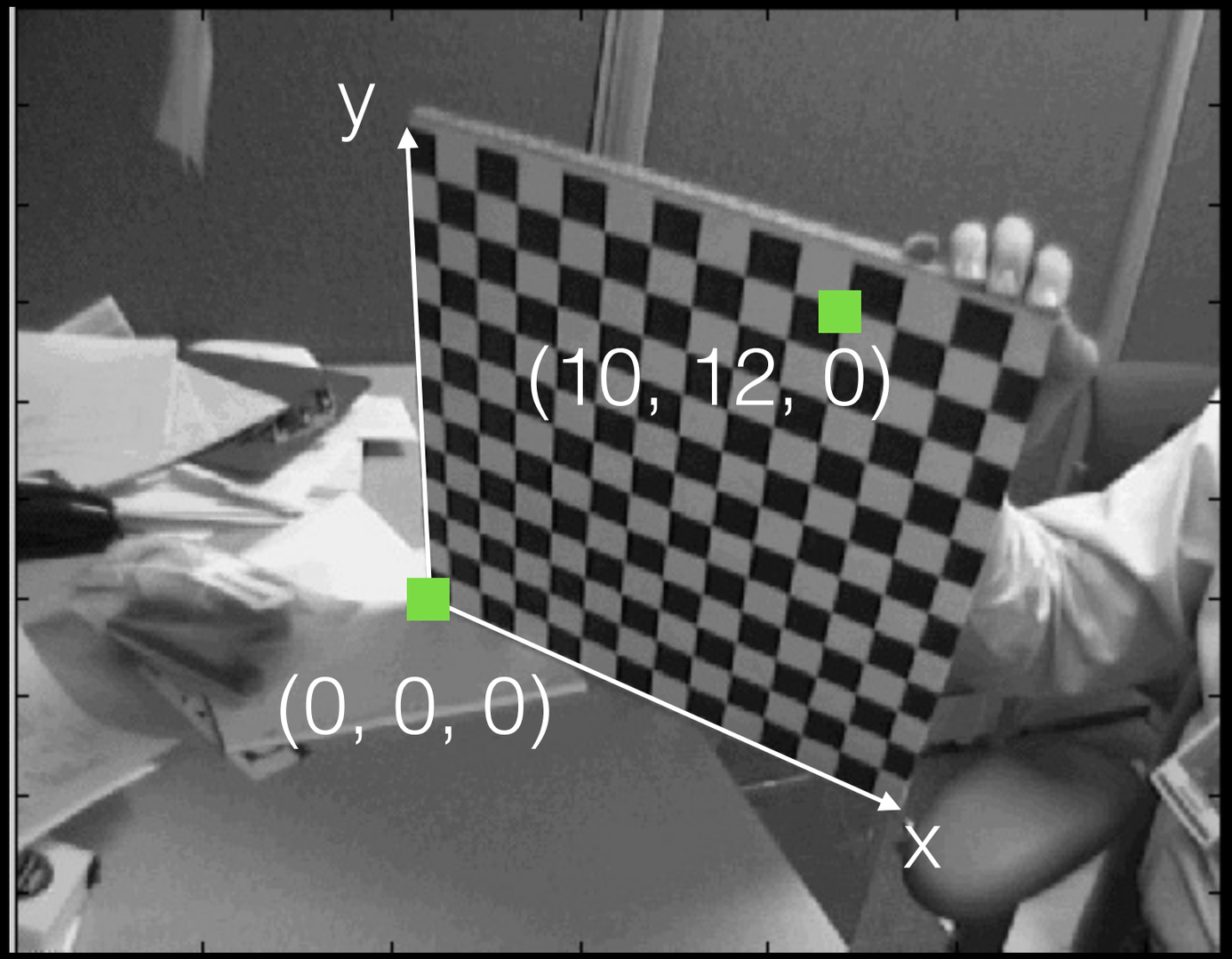
Camera Calibration



Camera Calibration



Camera Calibration



DLT Method

$$\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \cong \begin{bmatrix} m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{11} & m_{12} & m_{13} \\ m_{20} & m_{21} & m_{22} & m_{23} \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$

DLT Method

$$\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \cong \begin{bmatrix} m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{11} & m_{12} & m_{13} \\ m_{20} & m_{21} & m_{22} & m_{23} \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$

$$u_i = \frac{m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}}$$

$$v_i = \frac{m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}}$$

DLT Method

$$\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \cong \begin{bmatrix} m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{11} & m_{12} & m_{13} \\ m_{20} & m_{21} & m_{22} & m_{23} \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$

$$u_i = \frac{m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}}$$

$$v_i = \frac{m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}}$$

$$u_i(m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}) = m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03}$$
$$v_i(m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}) = m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13}$$

DLT Method

$$\begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \cong \begin{bmatrix} m_{00} & m_{01} & m_{02} & m_{03} \\ m_{10} & m_{11} & m_{12} & m_{13} \\ m_{20} & m_{21} & m_{22} & m_{23} \end{bmatrix} \begin{bmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{bmatrix}$$

$$u_i = \frac{m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}}$$

$$v_i = \frac{m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13}}{m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}}$$

$$u_i(m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}) = m_{00}X_i + m_{01}Y_i + m_{02}Z_i + m_{03}$$
$$v_i(m_{20}X_i + m_{21}Y_i + m_{22}Z_i + m_{23}) = m_{10}X_i + m_{11}Y_i + m_{12}Z_i + m_{13}$$

$$\begin{bmatrix} X_i & Y_i & Z_i & 1 & 0 & 0 & 0 & 0 & -u_iX_i & -u_iY_i & -u_iZ_i & -u_i \\ 0 & 0 & 0 & 0 & X_i & Y_i & Z_i & 1 & -v_iX_i & -v_iY_i & -v_iZ_i & -v_i \end{bmatrix} \begin{bmatrix} m_{00} \\ m_{01} \\ m_{02} \\ m_{03} \\ m_{10} \\ m_{11} \\ m_{12} \\ m_{13} \\ m_{20} \\ m_{21} \\ m_{22} \\ m_{23} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Question: Is a single plane enough?

$$\begin{bmatrix} X_i & Y_i & Z_i & 1 & 0 & 0 & 0 & 0 & -u_iX_i & -u_iY_i & -u_iZ_i & -u_i \\ 0 & 0 & 0 & 0 & X_i & Y_i & Z_i & 1 & -v_iX_i & -v_iY_i & -v_iZ_i & -v_i \end{bmatrix} \begin{bmatrix} m_{00} \\ m_{01} \\ m_{02} \\ m_{03} \\ m_{10} \\ m_{11} \\ m_{12} \\ m_{13} \\ m_{20} \\ m_{21} \\ m_{22} \\ m_{23} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Question: Is a single plane enough?

$$\begin{bmatrix} X_i & Y_i & Z_i & 1 & 0 & 0 & 0 & 0 & -u_iX_i & -u_iY_i & -u_iZ_i & -u_i \\ 0 & 0 & 0 & 0 & X_i & Y_i & Z_i & 1 & -v_iX_i & -v_iY_i & -v_iZ_i & -v_i \end{bmatrix} \begin{bmatrix} m_{00} \\ m_{01} \\ m_{02} \\ m_{03} \\ m_{10} \\ m_{11} \\ m_{12} \\ m_{13} \\ m_{20} \\ m_{21} \\ m_{22} \\ m_{23} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Assume plane is at Z = 0

(rotate and translate coordinates to make it so)

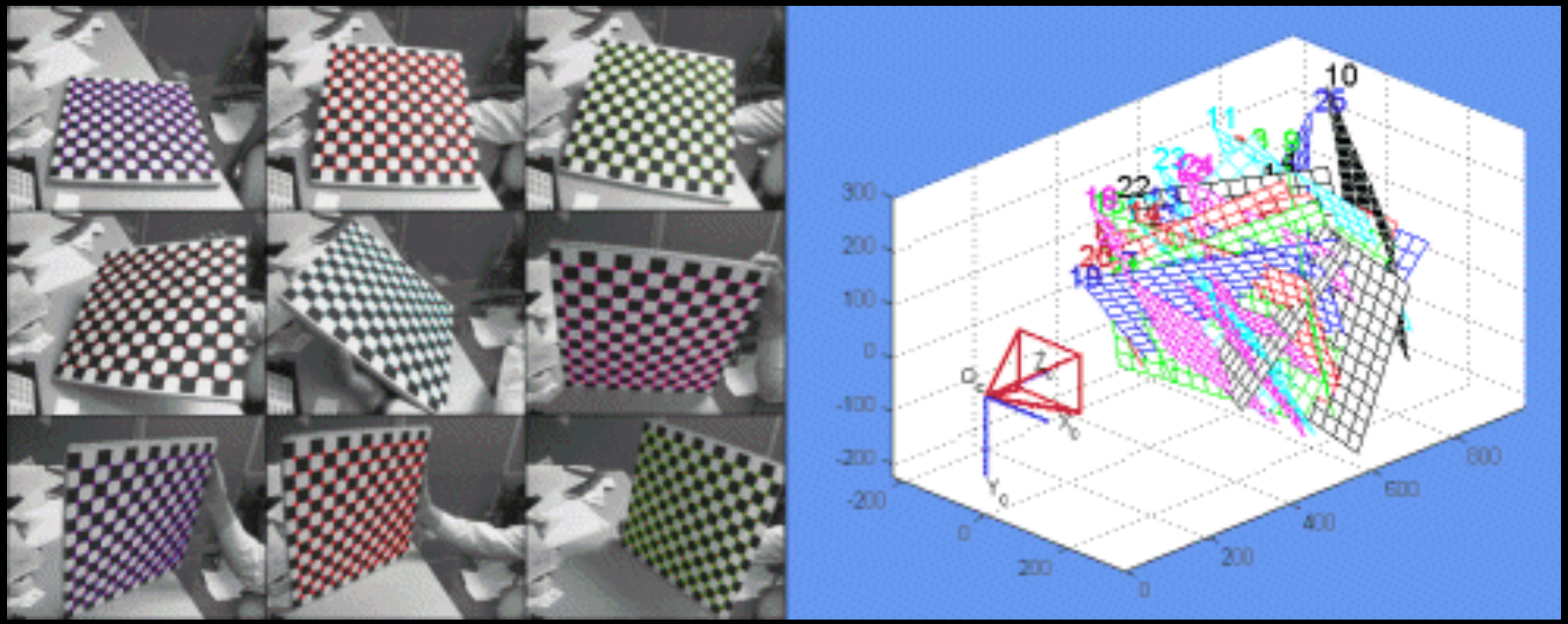
$$\begin{bmatrix} X_i & Y_i & \mathbf{0} & 1 & 0 & 0 & 0 & 0 & -u_iX_i & -u_iY_i & \mathbf{0} & -u_i \\ 0 & 0 & 0 & 0 & X_i & Y_i & \mathbf{0} & 1 & -v_iX_i & -v_iY_i & \mathbf{0} & -v_i \end{bmatrix} \begin{bmatrix} m_{00} \\ m_{01} \\ m_{02} \\ m_{03} \\ m_{10} \\ m_{11} \\ m_{12} \\ m_{13} \\ m_{20} \\ m_{21} \\ m_{22} \\ m_{23} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Question: Is a single plane enough?

$$\begin{bmatrix} X_i & Y_i & \mathbf{0} & 1 & 0 & 0 & \mathbf{0} & 0 & -u_iX_i & -u_iY_i & \mathbf{0} & -u_i \\ 0 & 0 & 0 & 0 & X_i & Y_i & \mathbf{0} & 1 & -v_iX_i & -v_iY_i & \mathbf{0} & -v_i \end{bmatrix} \begin{bmatrix} m_{00} \\ m_{01} \\ m_{02} \\ m_{03} \\ m_{10} \\ m_{11} \\ m_{12} \\ m_{13} \\ m_{20} \\ m_{21} \\ m_{22} \\ m_{23} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Columns are all 0
> Rank is at most 9

No, calibration target cannot be planar with DLT method.
But we can combine **many** planes.


300
200
100
0
-100
-200
-200
0
200
0
200
400
600
800
10
25
11
23
22
20
19
Oc
Zc
Xc
Yc

Non-Linear Method

- DLT method does not automatically give decomposition into extrinsics and intrinsics
- May wish to impose additional constraints on camera model (e.g. isotropic focal length, square pixels)
- Non-linearities such as radial distortion are not easily modeled with DLT

$$\begin{bmatrix} u_i w_i \\ v_i w_i \\ w_i \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \left[\begin{array}{ccc|c} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{array}\right] \begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} u_i w_i \\ v_i w_i \\ w_i \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \left[\begin{array}{ccc|c} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{array}\right] \begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix}$$

3D world-space point

$$\begin{bmatrix} u_i w_i \\ v_i w_i \\ w_i \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \left[\begin{array}{ccc|c} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{array}\right] \begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix}$$

Rotate and translate point into camera space

3D world-space point

$$\begin{bmatrix} u_i w_i \\ v_i w_i \\ w_i \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \left[\begin{array}{ccc|c} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{array}\right] \begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix}$$

Project point into image plane

Rotate and translate point into camera space

3D world-space point

$$\begin{bmatrix} u_i w_i \\ v_i w_i \\ w_i \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \left[\begin{array}{ccc|c} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{array}\right] \begin{bmatrix} x_i \\ y_i \\ z_i \\ 1 \end{bmatrix}$$

Project point into image plane

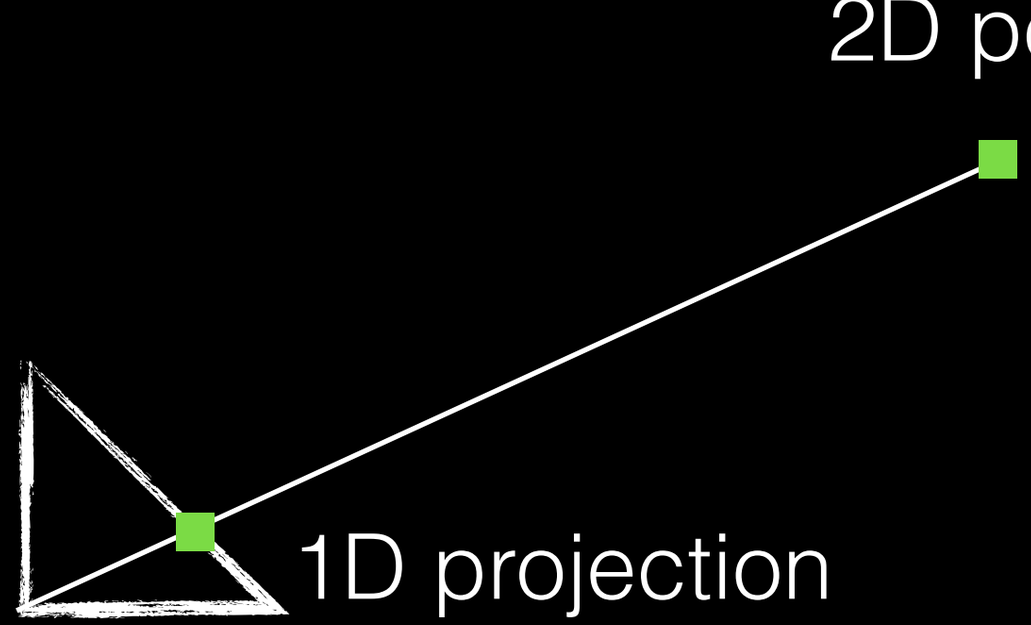
Rotate and translate point into camera space

3D world-space point

Let's work through a simpler 2D version

$$\begin{bmatrix} u_i w_i \\ w_i \end{bmatrix} = \begin{bmatrix} f & c \\ 0 & 1 \end{bmatrix} \left[\begin{array}{cc|c} \cos(\theta) & -\sin(\theta) & t_x \\ \sin(\theta) & \cos(\theta) & t_y \end{array}\right] \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} u_i w_i \\ w_i \end{bmatrix} = \begin{bmatrix} f & c \\ 0 & 1 \end{bmatrix} \left[\begin{array}{cc|c} \cos(\theta) & -\sin(\theta) & t_x \\ \sin(\theta) & \cos(\theta) & t_y \end{array}\right] \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$



$$\begin{bmatrix} u_i w_i \\ w_i \end{bmatrix} = \begin{bmatrix} f & c \\ 0 & 1 \end{bmatrix} \left[\begin{array}{cc|c} \cos(\theta) & -\sin(\theta) & t_x \\ \sin(\theta) & \cos(\theta) & t_y \end{array}\right] \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} u_i w_i \\ w_i \end{bmatrix} = \begin{bmatrix} f & c \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\theta)x_i - \sin(\theta)y_i + t_x \\ \sin(\theta)x_i + \cos(\theta)y_i + t_y \end{bmatrix}$$

$$\begin{bmatrix} u_i w_i \\ w_i \end{bmatrix} = \begin{bmatrix} f & c \\ 0 & 1 \end{bmatrix} \left[\begin{array}{cc|c} \cos(\theta) & -\sin(\theta) & t_x \\ \sin(\theta) & \cos(\theta) & t_y \end{array}\right] \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} u_i w_i \\ w_i \end{bmatrix} = \begin{bmatrix} f & c \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\theta)x_i - \sin(\theta)y_i + t_x \\ \sin(\theta)x_i + \cos(\theta)y_i + t_y \end{bmatrix}$$

$$\begin{bmatrix} u_i w_i \\ w_i \end{bmatrix} = \begin{bmatrix} f(\cos(\theta)x_i - \sin(\theta)y_i + t_x) + c(\sin(\theta)x_i + \cos(\theta)y_i + t_y) \\ \sin(\theta)x_i + \cos(\theta)y_i + t_y \end{bmatrix}$$

$$\begin{bmatrix} u_i w_i \\ w_i \end{bmatrix} = \begin{bmatrix} f & c \\ 0 & 1 \end{bmatrix} \left[\begin{array}{cc|c} \cos(\theta) & -\sin(\theta) & t_x \\ \sin(\theta) & \cos(\theta) & t_y \end{array}\right] \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} u_i w_i \\ w_i \end{bmatrix} = \begin{bmatrix} f & c \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\theta)x_i - \sin(\theta)y_i + t_x \\ \sin(\theta)x_i + \cos(\theta)y_i + t_y \end{bmatrix}$$

$$\begin{bmatrix} u_i w_i \\ w_i \end{bmatrix} = \begin{bmatrix} f(\cos(\theta)x_i - \sin(\theta)y_i + t_x) + c(\sin(\theta)x_i + \cos(\theta)y_i + t_y) \\ \sin(\theta)x_i + \cos(\theta)y_i + t_y \end{bmatrix}$$

$$u_i = \frac{f(\cos(\theta)x_i - \sin(\theta)y_i + t_x) + c(\sin(\theta)x_i + \cos(\theta)y_i + t_y)}{\sin(\theta)x_i + \cos(\theta)y_i + t_y}$$

$$h(f, c, \theta, t_x, t_y, x_i, y_i) = \frac{f(\cos(\theta)x_i - \sin(\theta)y_i + t_x) + c(\sin(\theta)x_i + \cos(\theta)y_i + t_y)}{\sin(\theta)x_i + \cos(\theta)y_i + t_y}$$

$$h(f, c, \theta, t_x, t_y, x_i, y_i) = \frac{f(\cos(\theta)x_i - \sin(\theta)y_i + t_x) + c(\sin(\theta)x_i + \cos(\theta)y_i + t_y)}{\sin(\theta)x_i + \cos(\theta)y_i + t_y}$$

$$L(f, c, \theta, t_x, t_y) = \sum_i (u_i - h(f, c, \theta, t_x, t_y, x_i, y_i))^2$$

$$h(f, c, \theta, t_x, t_y, x_i, y_i) = \frac{f(\cos(\theta)x_i - \sin(\theta)y_i + t_x) + c(\sin(\theta)x_i + \cos(\theta)y_i + t_y)}{\sin(\theta)x_i + \cos(\theta)y_i + t_y}$$

$$L(f, c, \theta, t_x, t_y) = \sum_i (u_i - h(f, c, \theta, t_x, t_y, x_i, y_i))^2$$

$$\underset{f,c,\theta,t_x,t_y}{\operatorname{argmin}} \; L(f, c, \theta, t_x, t_y)$$

$$h(f, c, \theta, t_x, t_y, x_i, y_i) = \frac{f(\cos(\theta)x_i - \sin(\theta)y_i + t_x) + c(\sin(\theta)x_i + \cos(\theta)y_i + t_y)}{\sin(\theta)x_i + \cos(\theta)y_i + t_y}$$

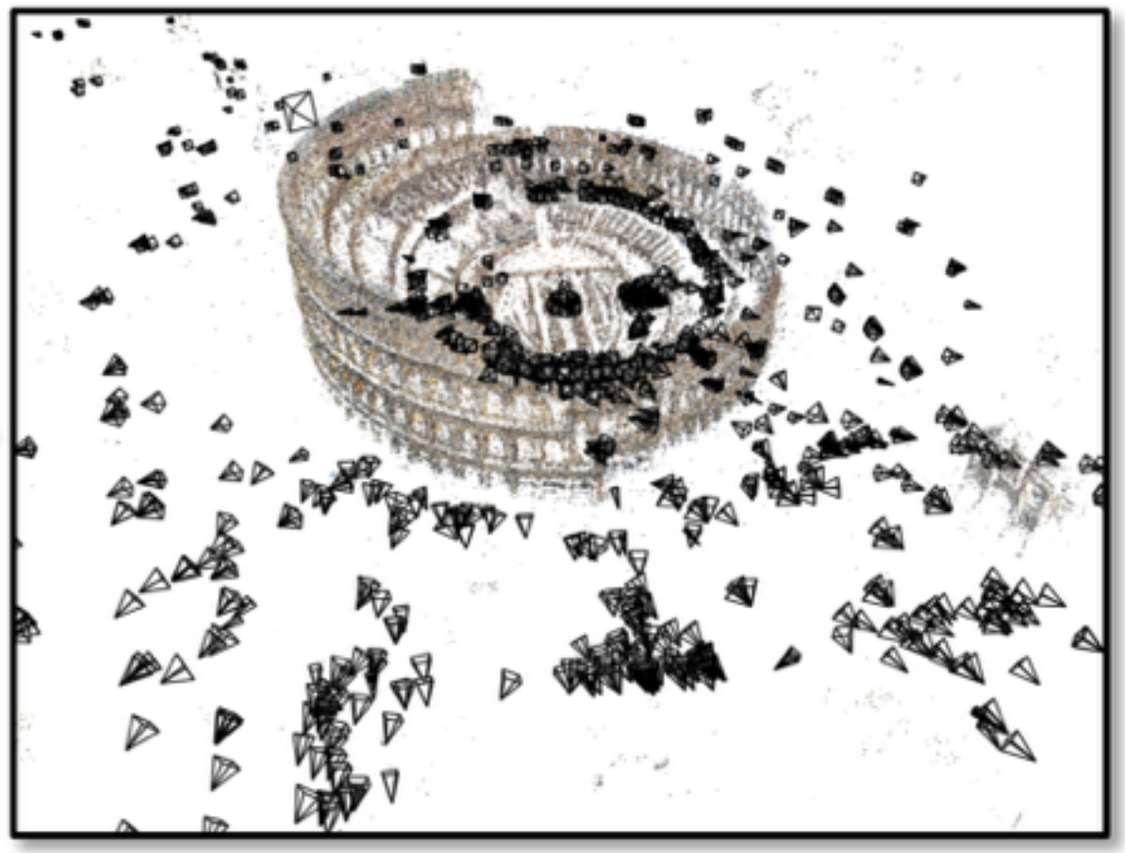
$$L(f, c, \theta, t_x, t_y) = \sum_i (u_i - h(f, c, \theta, t_x, t_y, x_i, y_i))^2$$

$$\underset{f,c,\theta,t_x,t_y}{\operatorname{argmin}} \; L(f, c, \theta, t_x, t_y)$$

Apply non-linear optimization method.
Exercise: Derive $\frac{\partial L}{\partial f}, \frac{\partial L}{\partial c}, \frac{\partial L}{\partial \theta}, \frac{\partial L}{\partial t_x}, \frac{\partial L}{\partial t_y}$

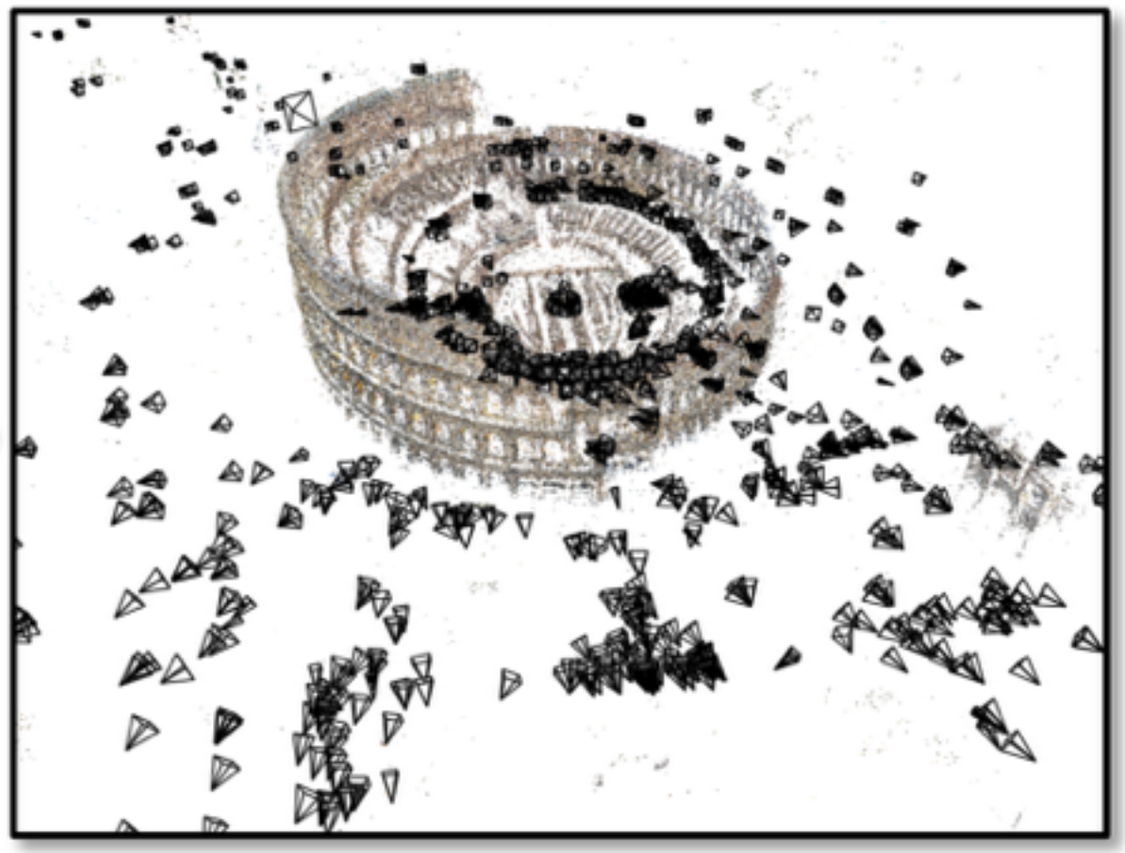
What if we don’t have a target?

What if we don’t have a target?



The world is our calibration target!

What if we don’t have a target?



The world is our calibration target!
But we don’t know the position of all points in the world.🤔

Structure from Motion

- Key goals of SfM:
  - Use **approximate** camera calibrations to match features and triangulate **approximate** 3D points
  - Use **approximate** 3D points to improve **approximate** camera calibrations
- Chicken-and-egg problem
  - Can extend and use our non-linear optimization framework
  - Requires a good initialization

SfM building blocks

- What do we need from our CV toolbox?
  - Keypoint detection
  - Descriptor matching
  - F-matrix estimation
  - Ray triangulation
  - Camera projection
  - Non-linear optimization
- Useful metadata
  - Focal length guess (EXIF tags)

1

Given:

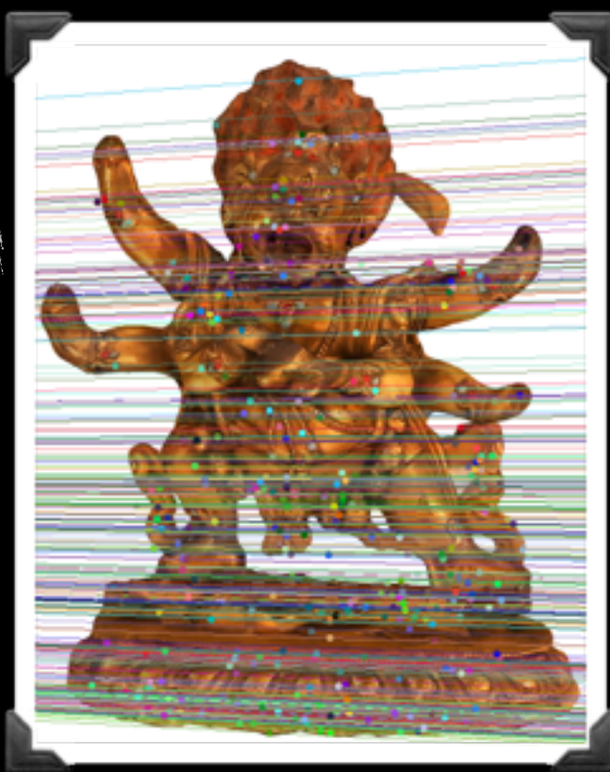
- Images 1 and 2
- Focal length guesses

2

1

1. Compute feature matches and F-matrix

2

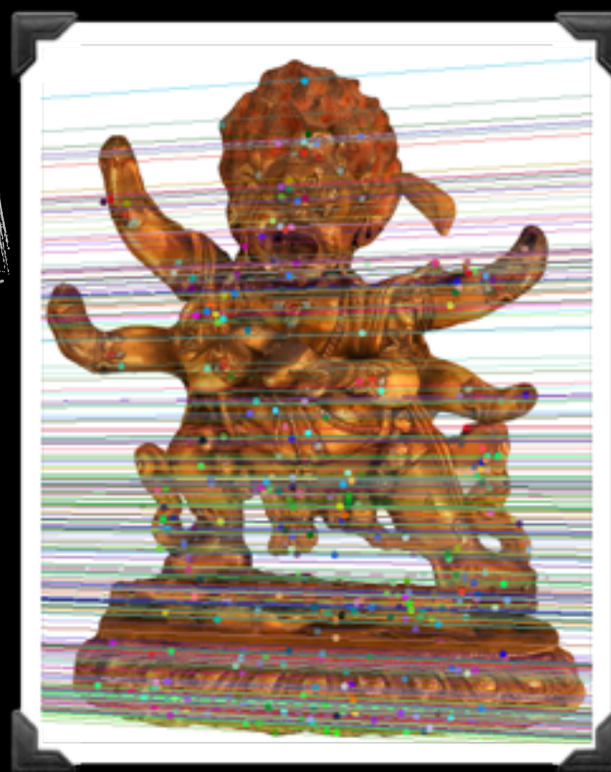


1

2. Use approx K's to get E-matrix

$E = K_2^T F K_1$

2

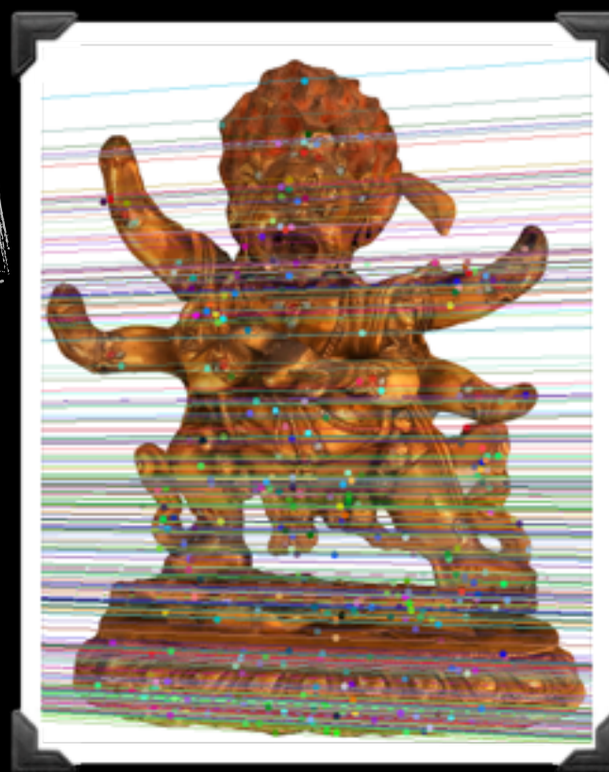


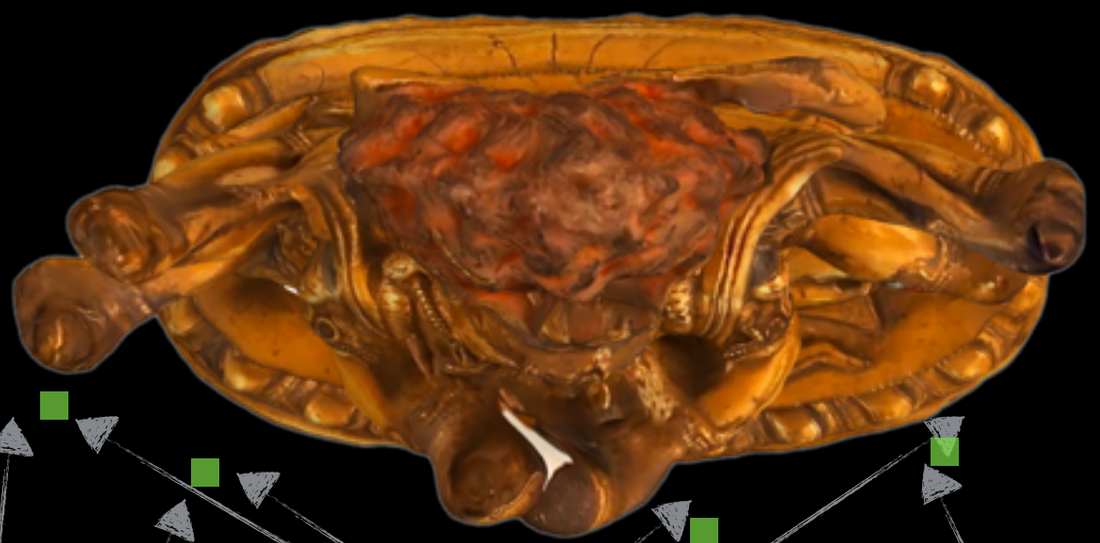
1

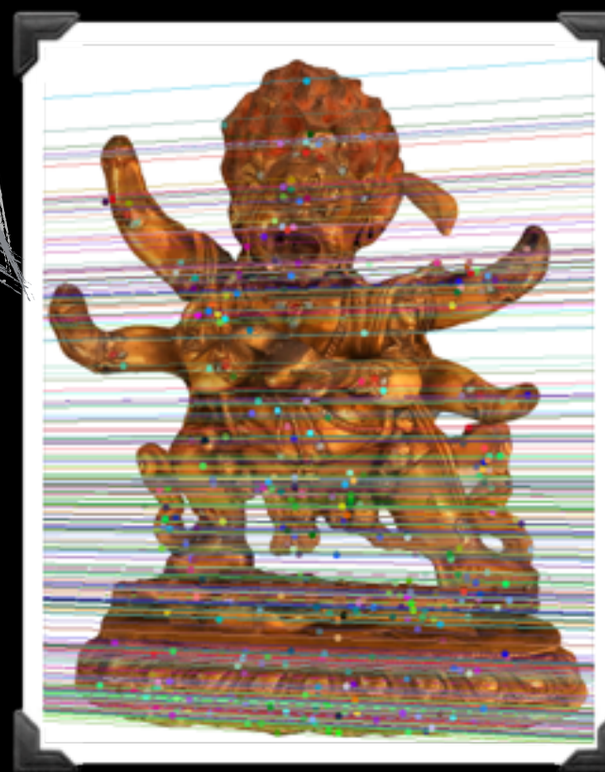
3. Decompose E into relative pose

$E = R[t]_x$

2



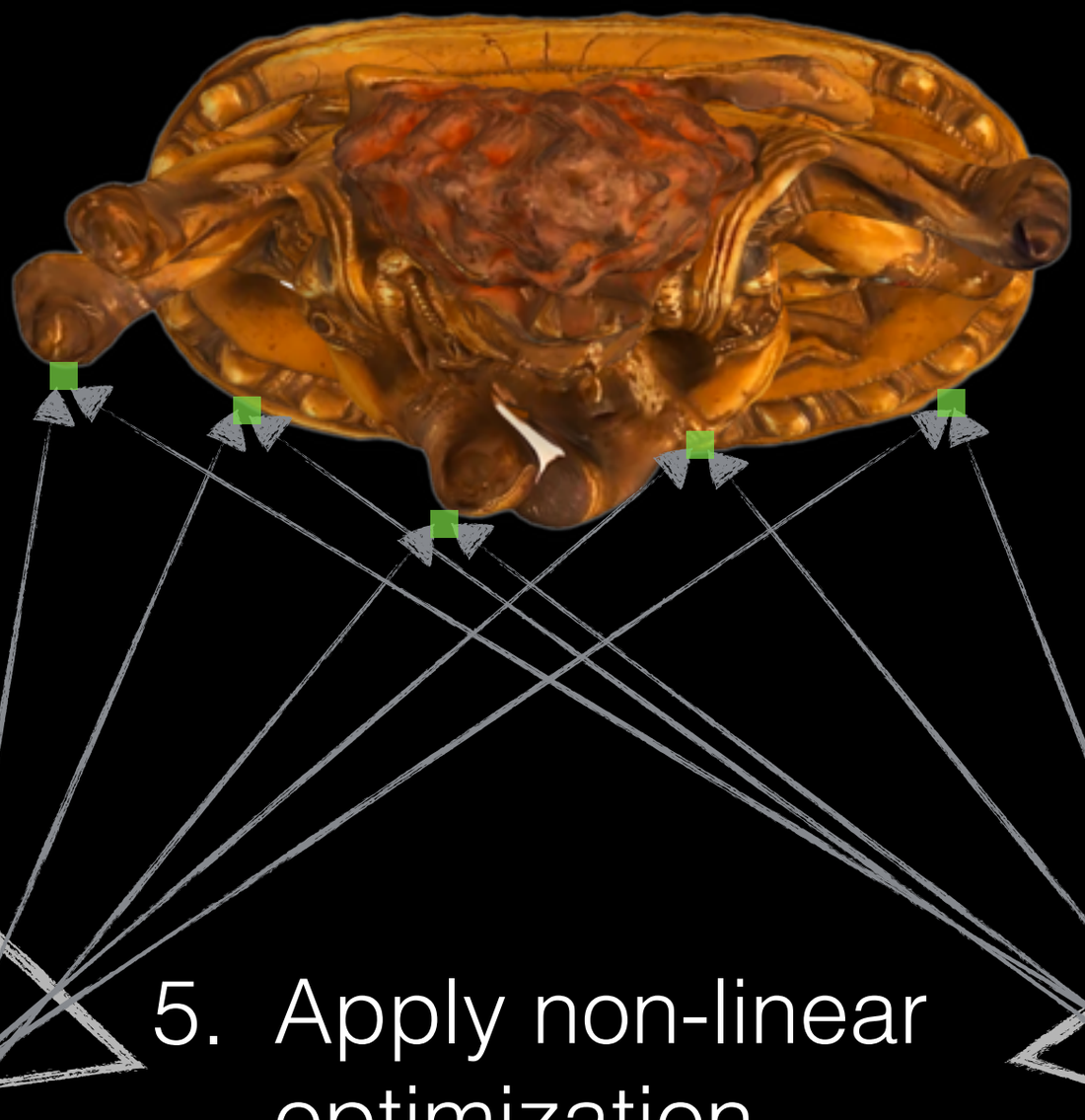




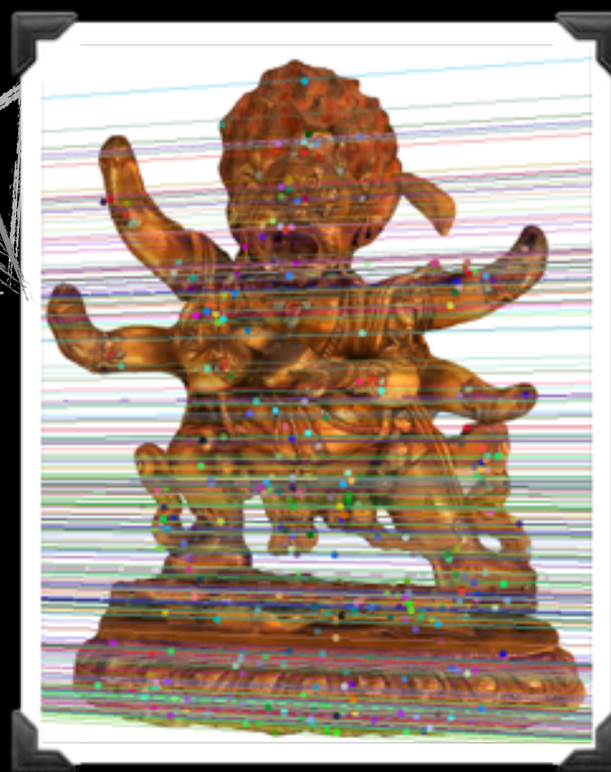
1

2

4. Triangulate features



5. Apply non-linear optimization



$$h(f, c, \theta, t_x, t_y, x_i, y_i) = \frac{f(\cos(\theta)x_i - \sin(\theta)y_i + t_x) + c(\sin(\theta)x_i + \cos(\theta)y_i + t_y)}{\sin(\theta)x_i + \cos(\theta)y_i + t_y}$$

$$L(f, c, \theta, t_x, t_y) = \sum_i (u_i - h(f, c, \theta, t_x, t_y, x_i, y_i))^2$$

$$\underset{f,c,\theta,t_x,t_y}{\operatorname{argmin}} \; L(f, c, \theta, t_x, t_y)$$

$$h(f, c, \theta, t_x, t_y, x_i, y_i) = \frac{f(\cos(\theta)x_i - \sin(\theta)y_i + t_x) + c(\sin(\theta)x_i + \cos(\theta)y_i + t_y)}{\sin(\theta)x_i + \cos(\theta)y_i + t_y}$$

$$L(f, c, \theta, t_x, t_y, (x_1, y_1), \ldots, (x_n, y_n)) = \sum_i (u_i - h(f, c, \theta, t_x, t_y, x_i, y_i))^2$$

$$\underset{f,c,\theta,t_x,t_y,(x_1,y_1),\ldots,(x_n,y_n)}{\operatorname{argmin}} \quad L(f, c, \theta, t_x, t_y, (x_1, y_1), \ldots, (x_n, y_n))$$

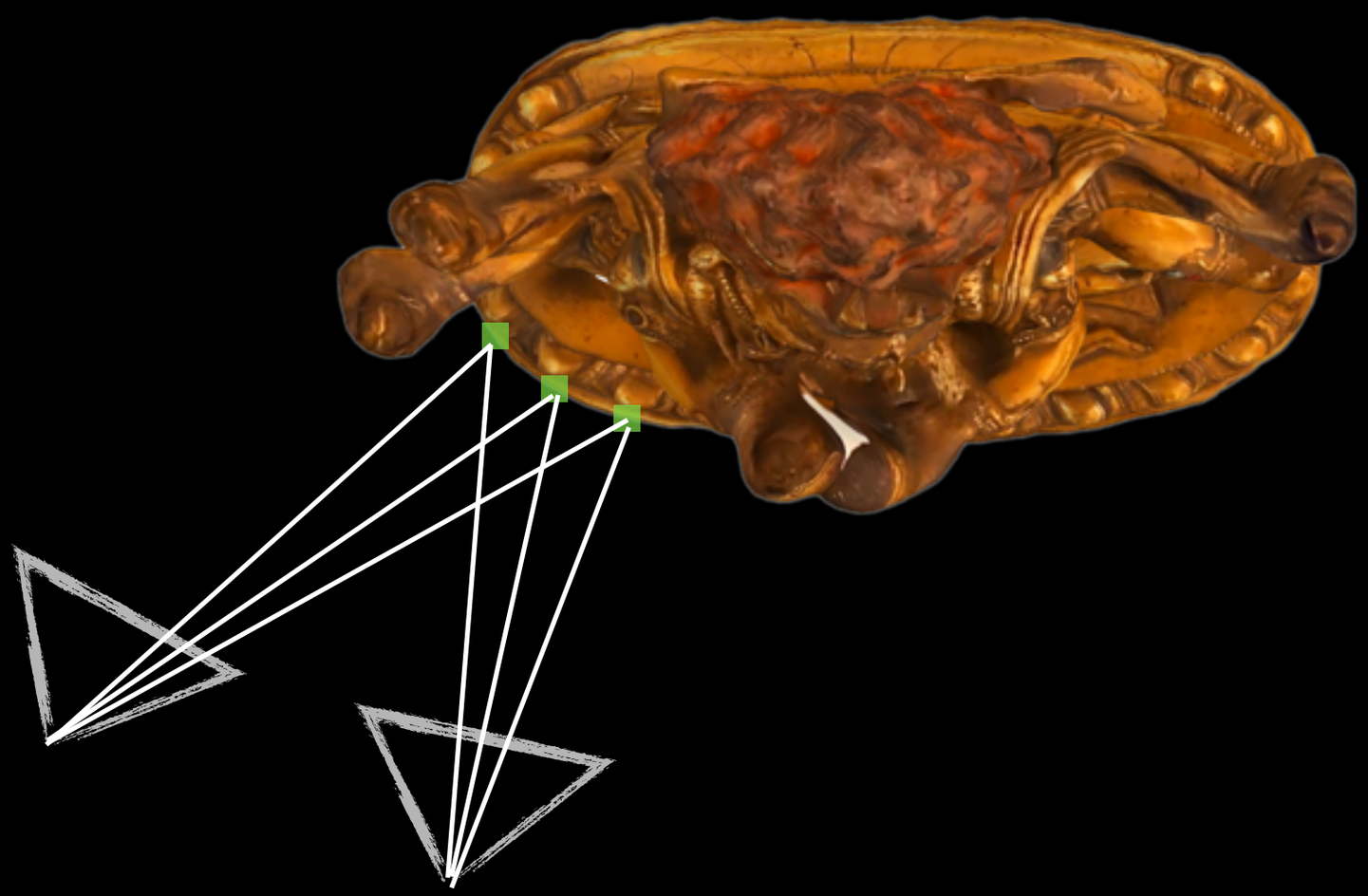
Doesn’t make sense for 1 camera

$$h(f, c, \theta, t_x, t_y, x_i, y_i) = \frac{f(\cos(\theta)x_i - \sin(\theta)y_i + t_x) + c(\sin(\theta)x_i + \cos(\theta)y_i + t_y)}{\sin(\theta)x_i + \cos(\theta)y_i + t_y}$$

$$L(K_1, \ldots, K_m, (x_1, y_1), \ldots, (x_n, y_n)) = \sum_i \sum_j w_{i,j}(u_i - h(K_j, (x_i, y_i)))^2$$

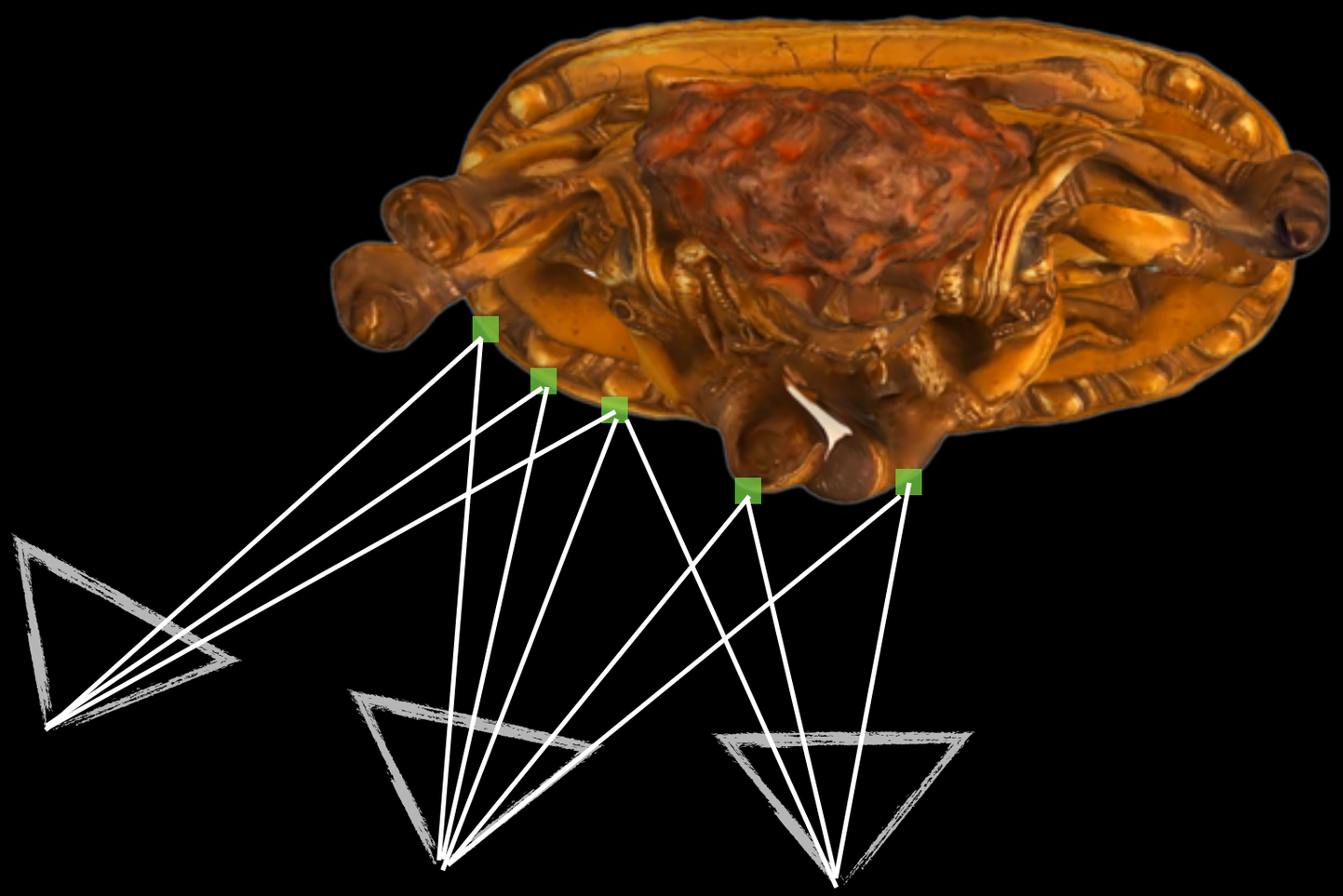
$$\underset{K_1,\ldots,K_m,(x_1,y_1),\ldots,(x_n,y_n)}{\operatorname{argmin}} L(K_1, \ldots, K_m, (x_1, y_1), \ldots, (x_n, y_n))$$

Called **Bundle Adjustment**



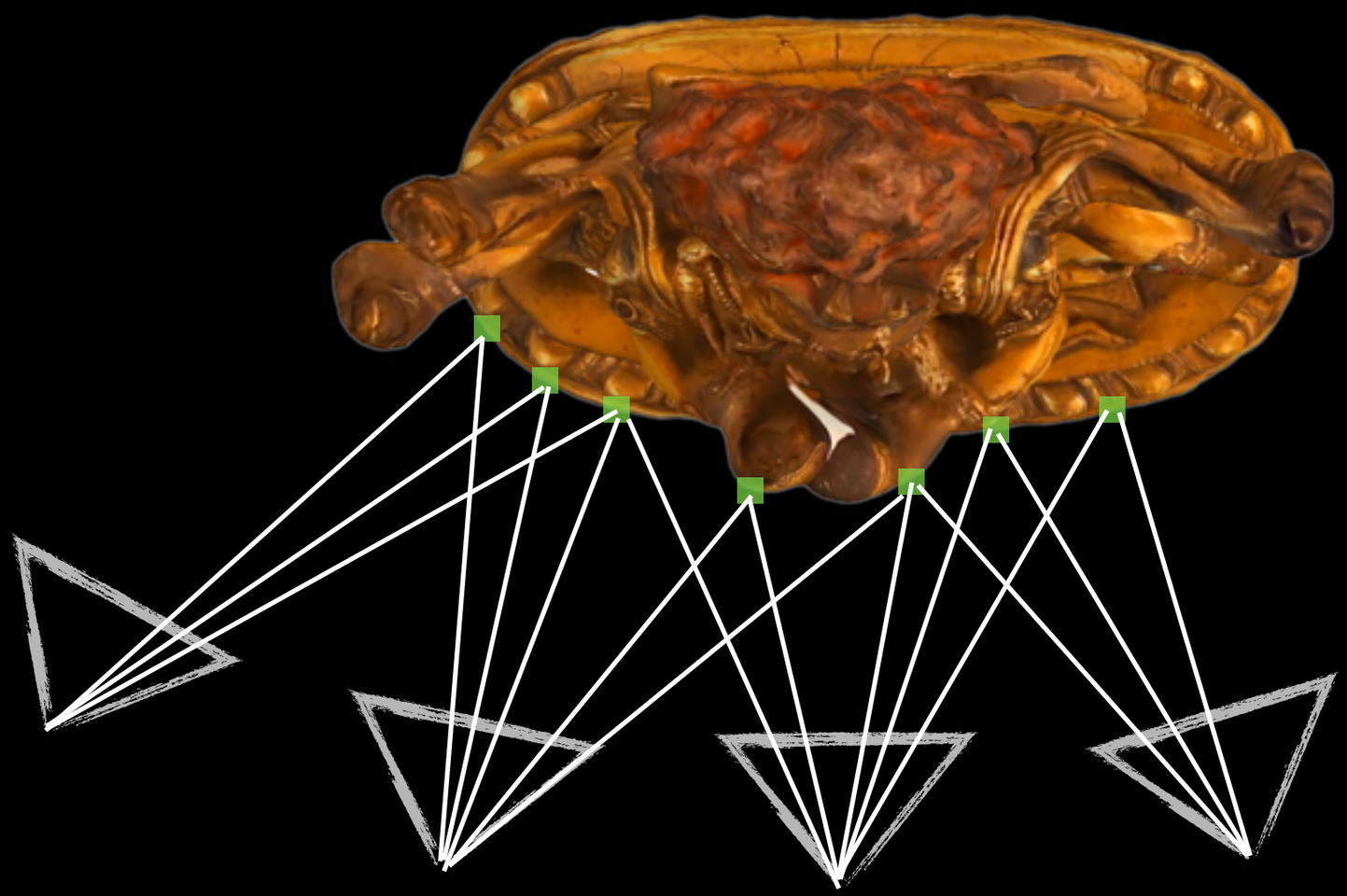
Camera sets can be incrementally built up

Essential matrix



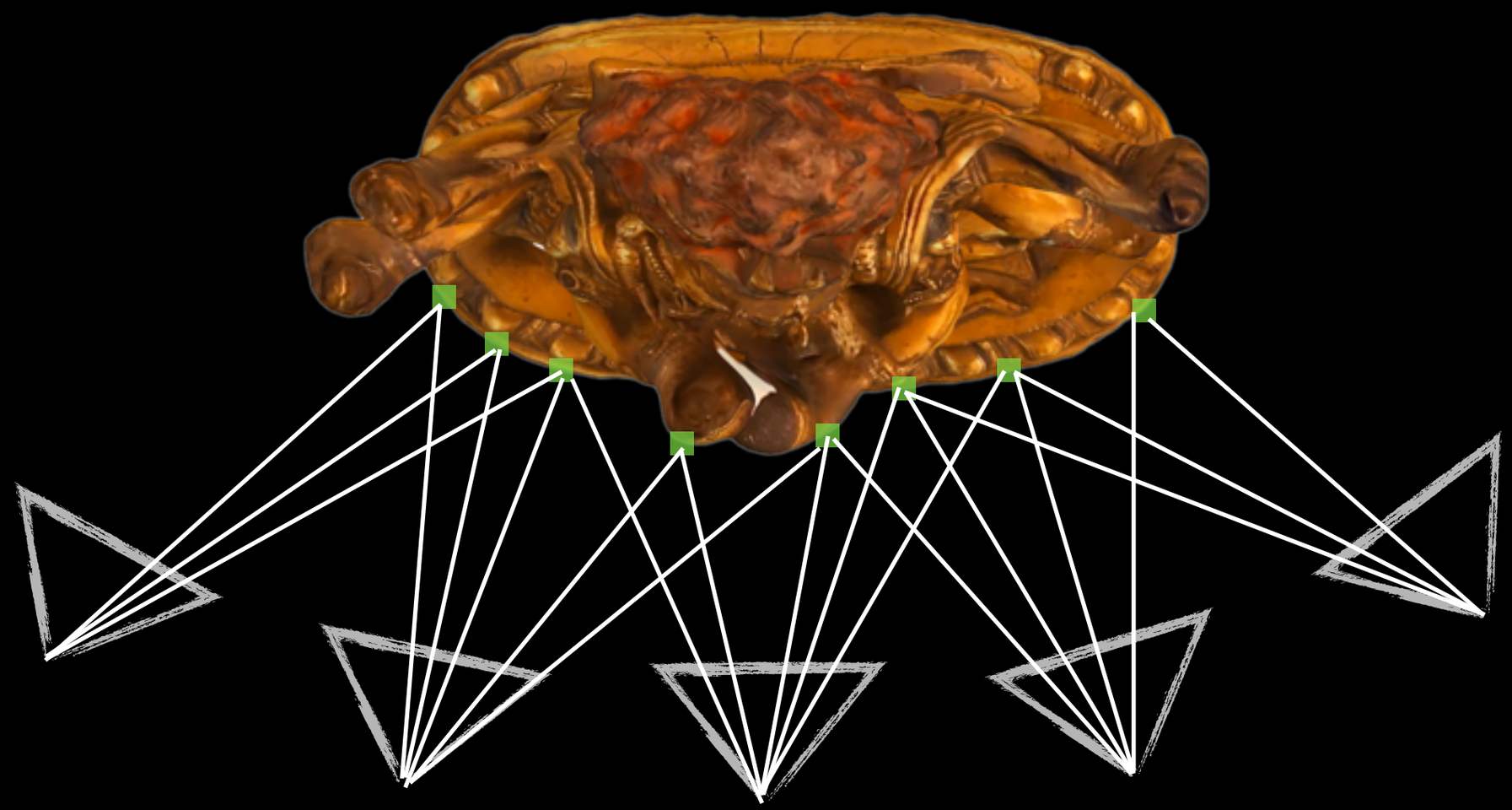
Camera sets can be incrementally built up

Perspective n Point method



Camera sets can be incrementally built up

Perspective n Point method

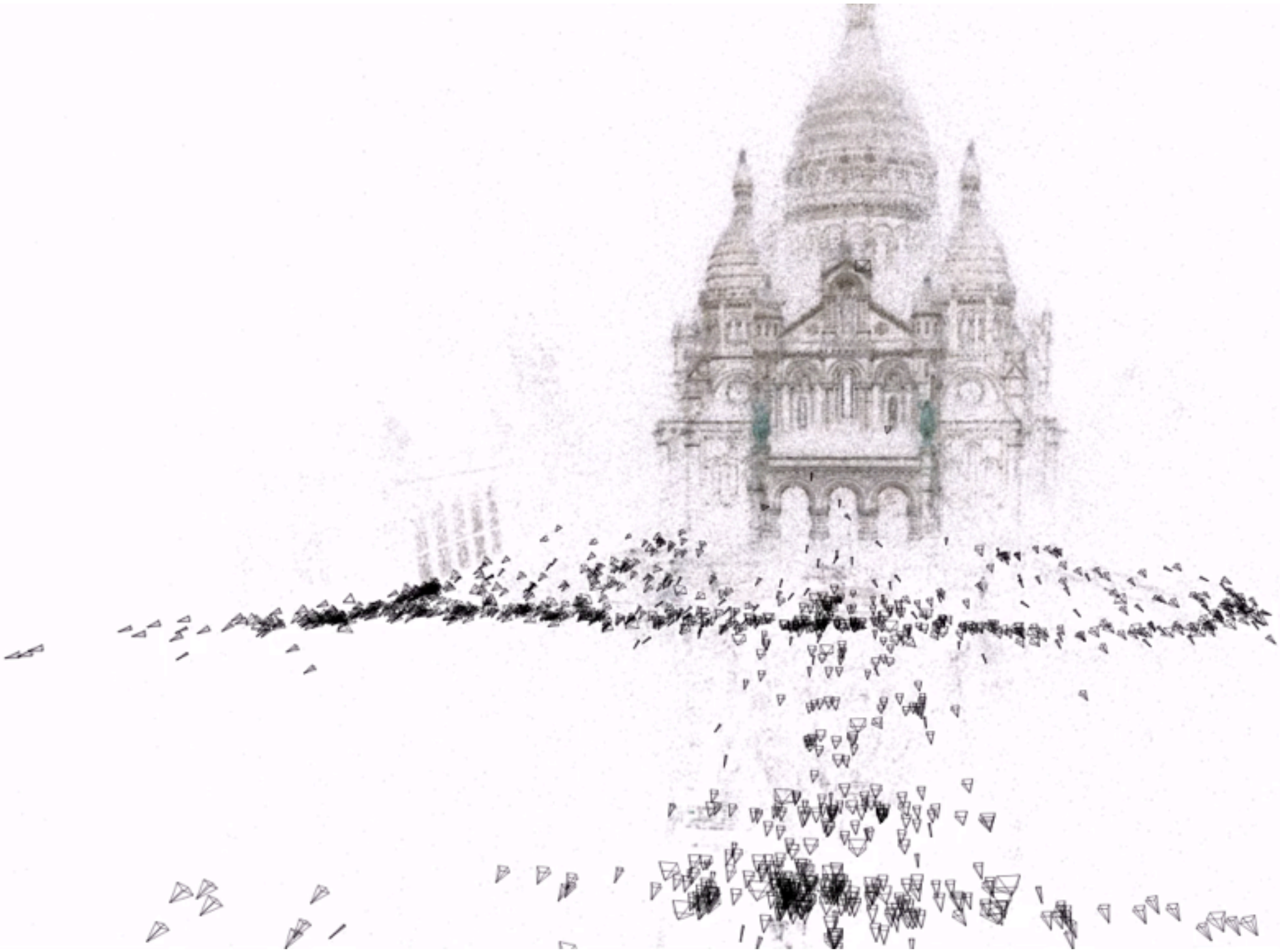


Camera sets can be incrementally built up

Perspective n Point method

Dubrovnik - Incremental Bundle Adjustment

Dubrovnik



Sacré-Cœur

SfM Ambiguities

$$x = PX$$

SfM Ambiguities

$$x = PX$$

$$x = (PQ)(Q^{-1}X)$$

SfM Ambiguities

$$x = PX$$

$$x = (PQ)(Q^{-1}X)$$

$$x = K(TQ)(Q^{-1}X)$$

SfM Ambiguities

$$x = PX$$

$$x = (PQ)(Q^{-1}X)$$

$$x = K(TQ)(Q^{-1}X)$$

T is a rigid body transformation
If we want TQ to be a RBT, then Q could be an RBT

> If we rotate and translate all the points, everything works out if we rotate and translate all the cameras.

SfM Ambiguities

$$x = PX$$

$$x = (PS^{-1})(SX)$$

SfM Ambiguities

$$x = PX$$

$$x = (PS^{-1})(SX)$$

$$x = K(TS^{-1})(SX)$$

SfM Ambiguities

$$x = PX$$

$$x = (PS^{-1})(SX)$$

$$x = K(TS^{-1})(SX)$$

$$x = K(S^{-1}TT')(SX)$$

$$x = (KS^{-1})(TT')(SX)$$

SfM Ambiguities

$$x = PX$$

$$x = (PS^{-1})(SX)$$

$$x = K(TS^{-1})(SX)$$

$$x = K(S^{-1}TT')(SX)$$

$$x = (KS^{-1})(TT')(SX)$$

$$x = (S^{-1}K)(TT')(SX)$$

$$Sx = K(TT')(SX)$$

SfM Ambiguities

$$Sx = K(TT')(SX)$$

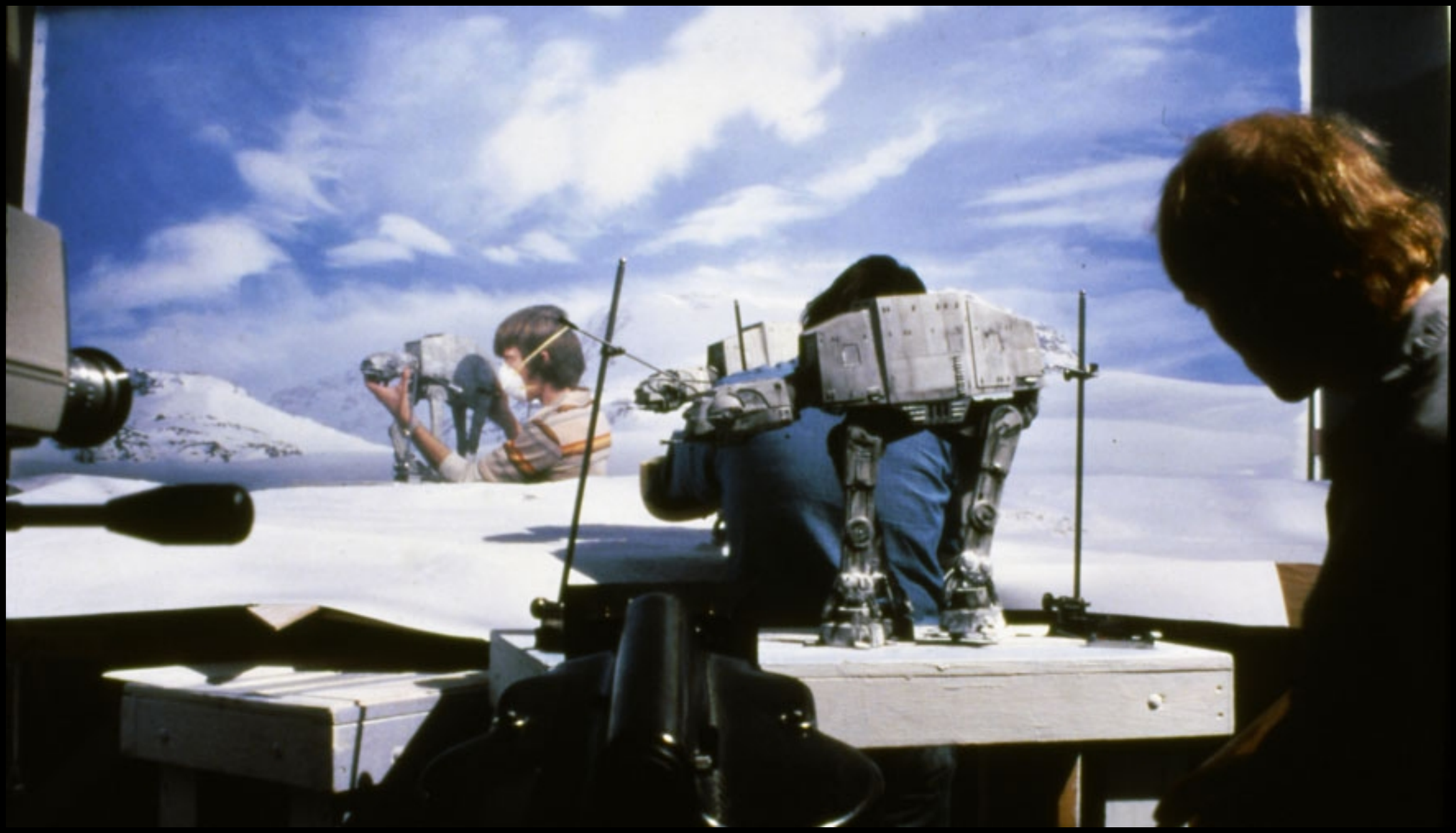
$$Sx = \begin{bmatrix} suw \\ svw \\ sw \end{bmatrix} \cong \begin{bmatrix} uw \\ vw \\ w \end{bmatrix} = x$$

SfM Ambiguities

$$Sx = K(TT')(SX)$$

$$Sx = \begin{bmatrix} suw \\ svw \\ sw \end{bmatrix} \cong \begin{bmatrix} uw \\ vw \\ w \end{bmatrix} = x$$

> If we scale all the points, everything works out if we move the camera positions.



SfM Ambiguities

$$x = PX$$

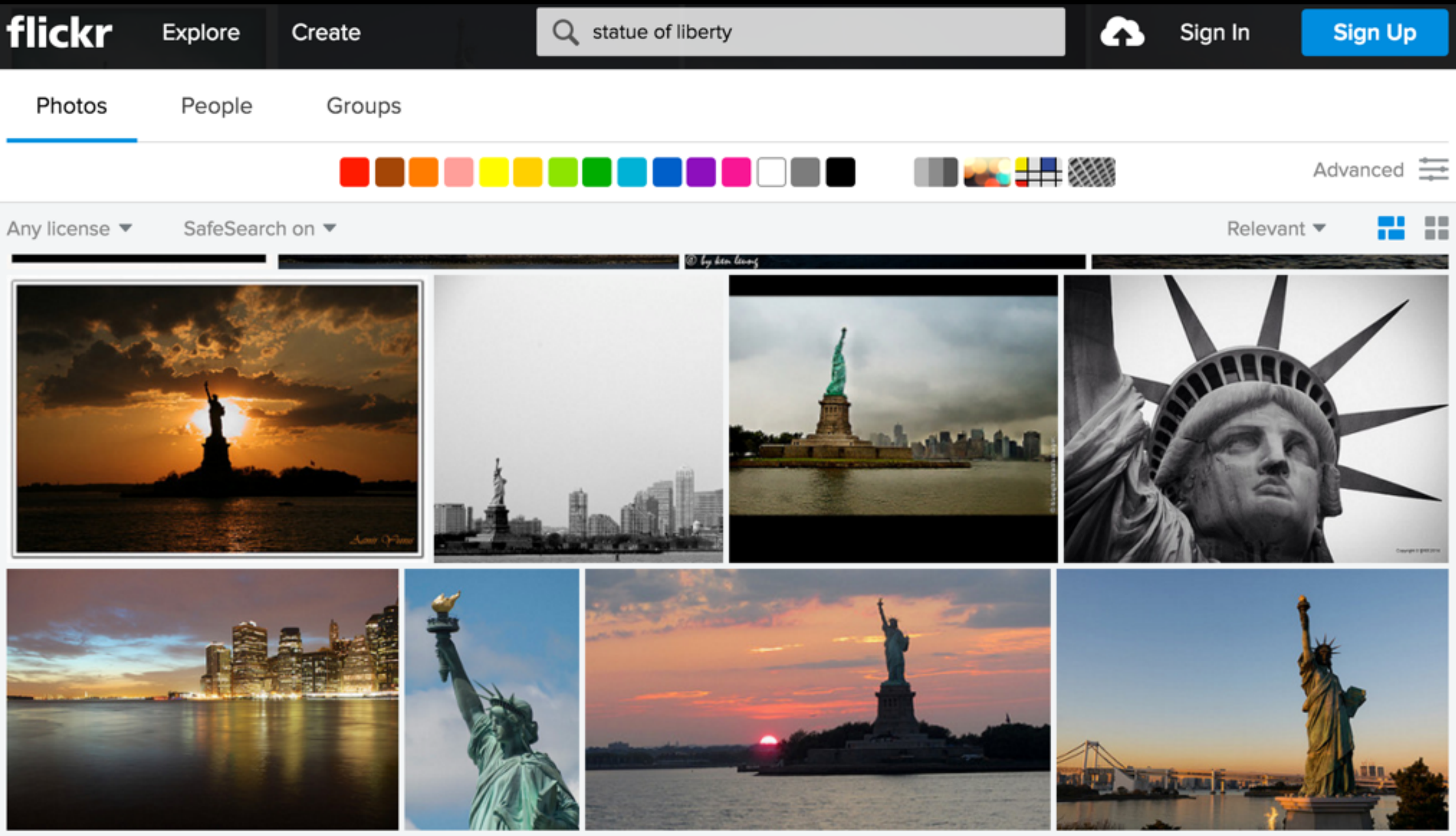
$$x = (PQ)(Q^{-1}X)$$

In this case Q is a general similarity transform.

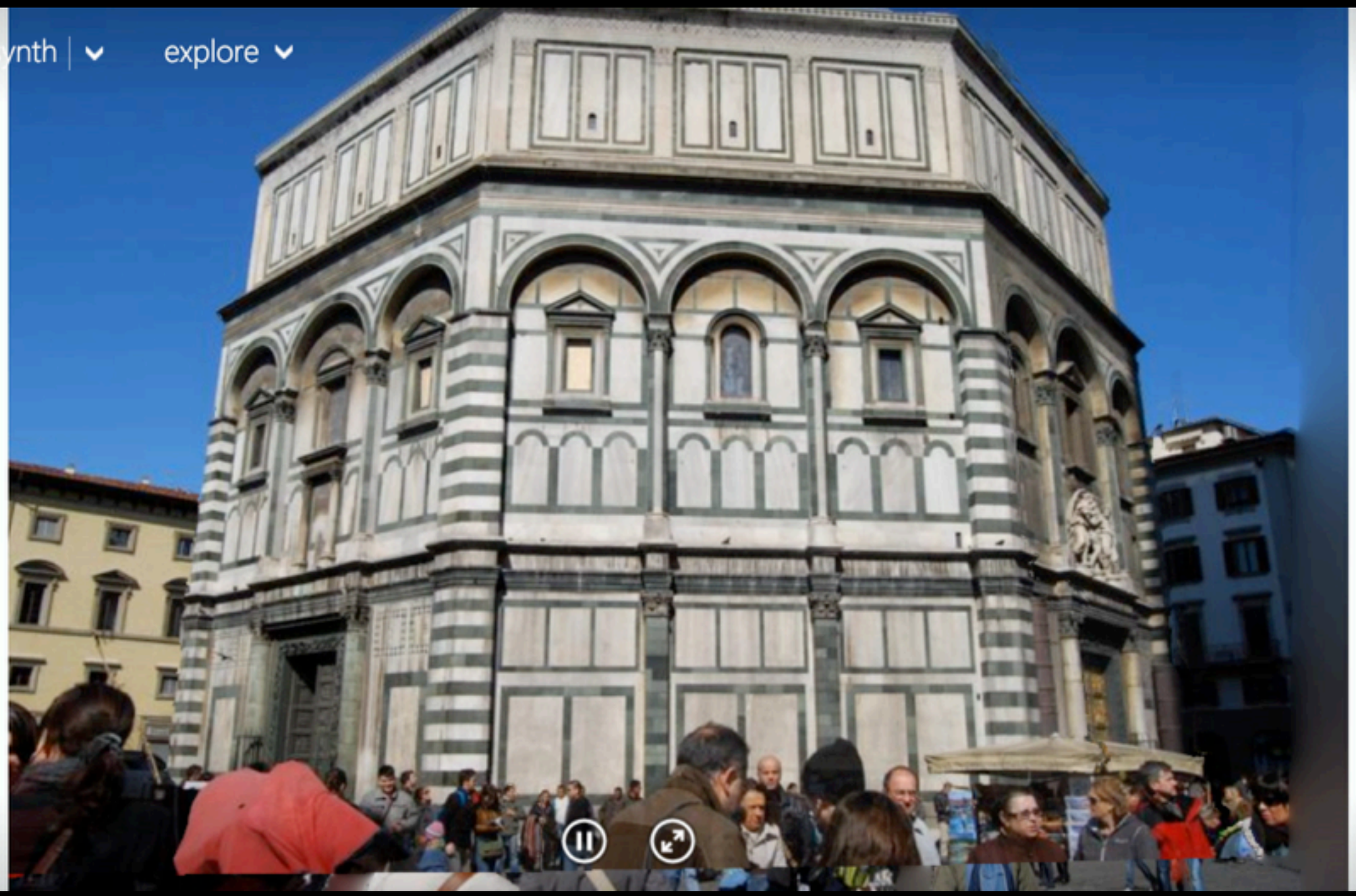
We resolve the ambiguity often by placing one camera at the origin facing some direction and a second camera at fixed offset from the first.

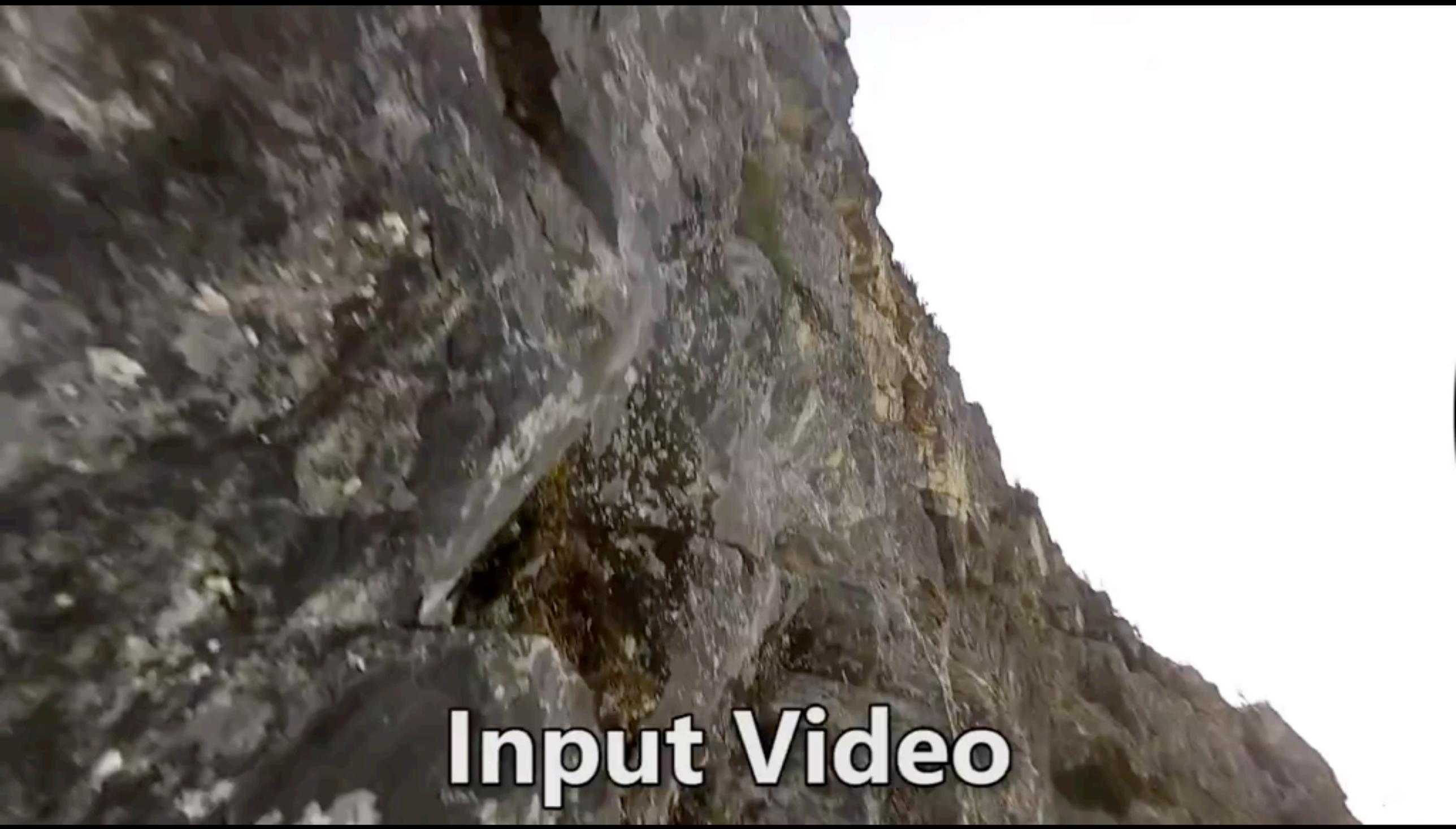
Applications

Internet-scale 3D



Snavely, et. al. Finding Paths through the World's Photos. SIGGRAPH 2008.





Microsoft Hyperlapse