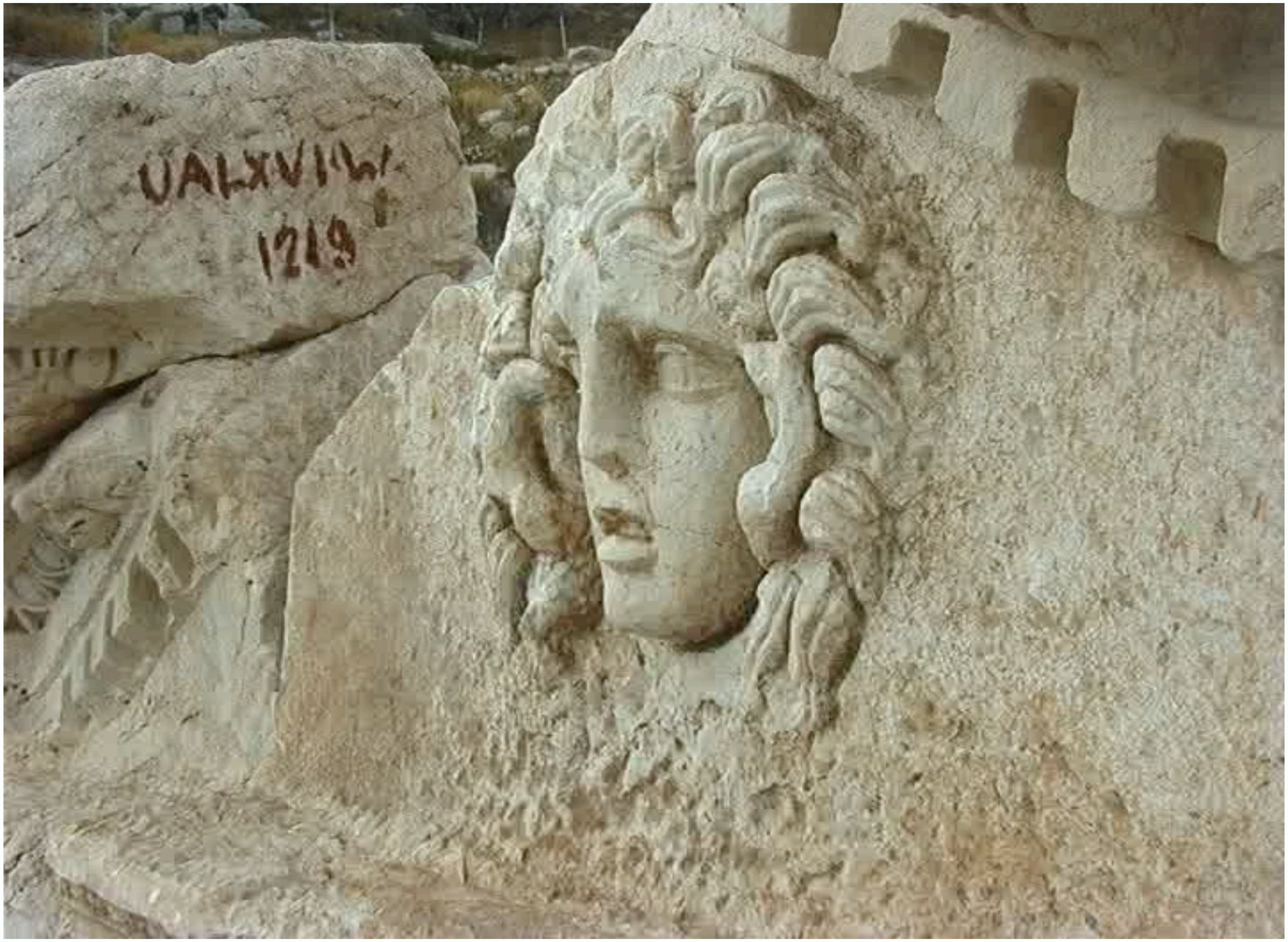
Basic SFM and SLAM

D.A. Forsyth, UIUC

Camera and structure from motion

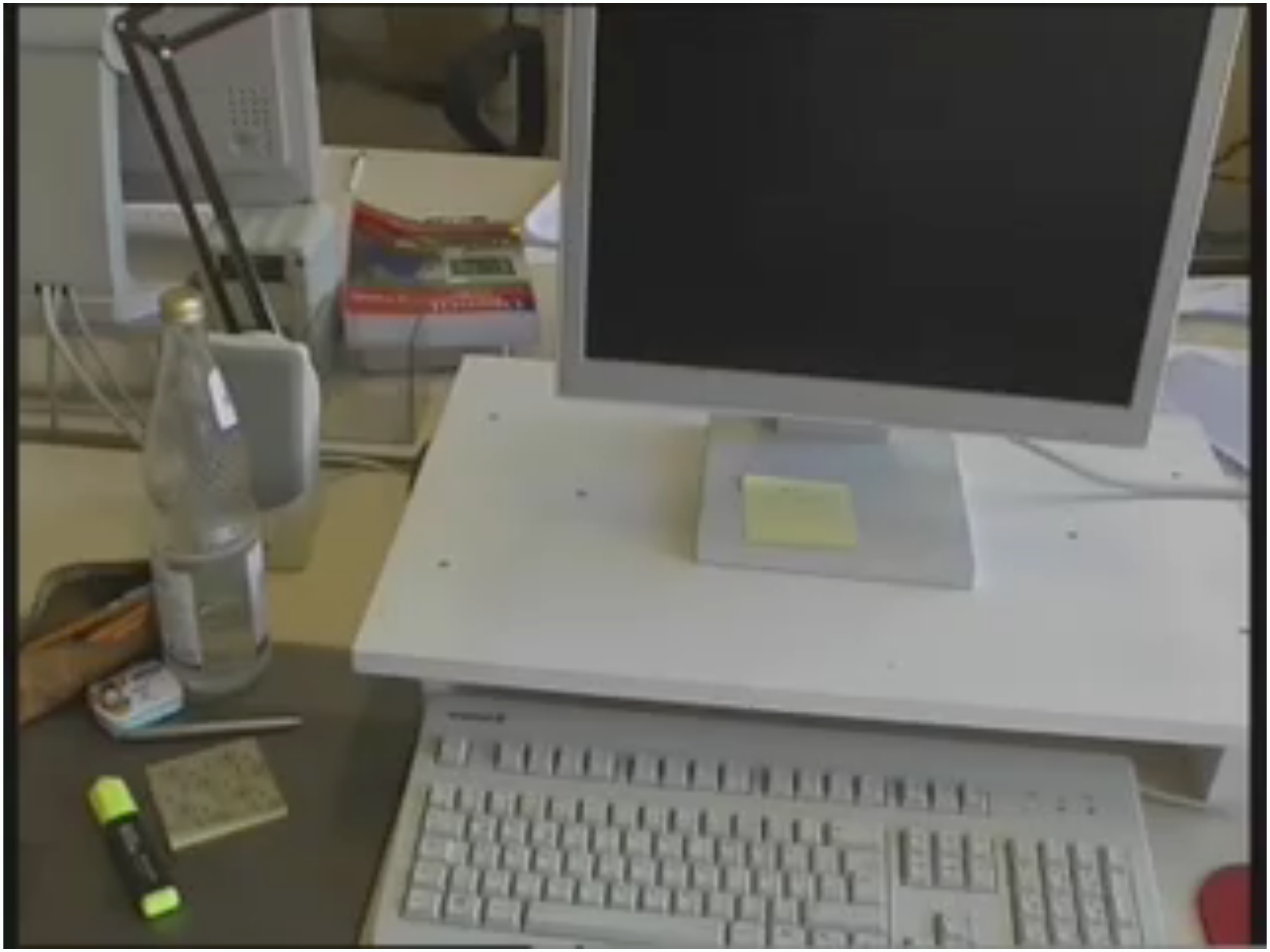
- Assume:
  - a moving camera views a static scene
  - the camera is orthographic OR
    - weak perspective applies with one scale for all
  - all points can be seen in all views AND all correspondences are known
- Can get:
  - the positions of all points in the scene
  - the configuration of each camera
- Applications
  - Reconstruction: Build a 3D model out of the reconstructed points
  - Mapping: Use the camera information to figure out where you went
  - Object insertion: Render a 3D model using the cameras, then composite the videos

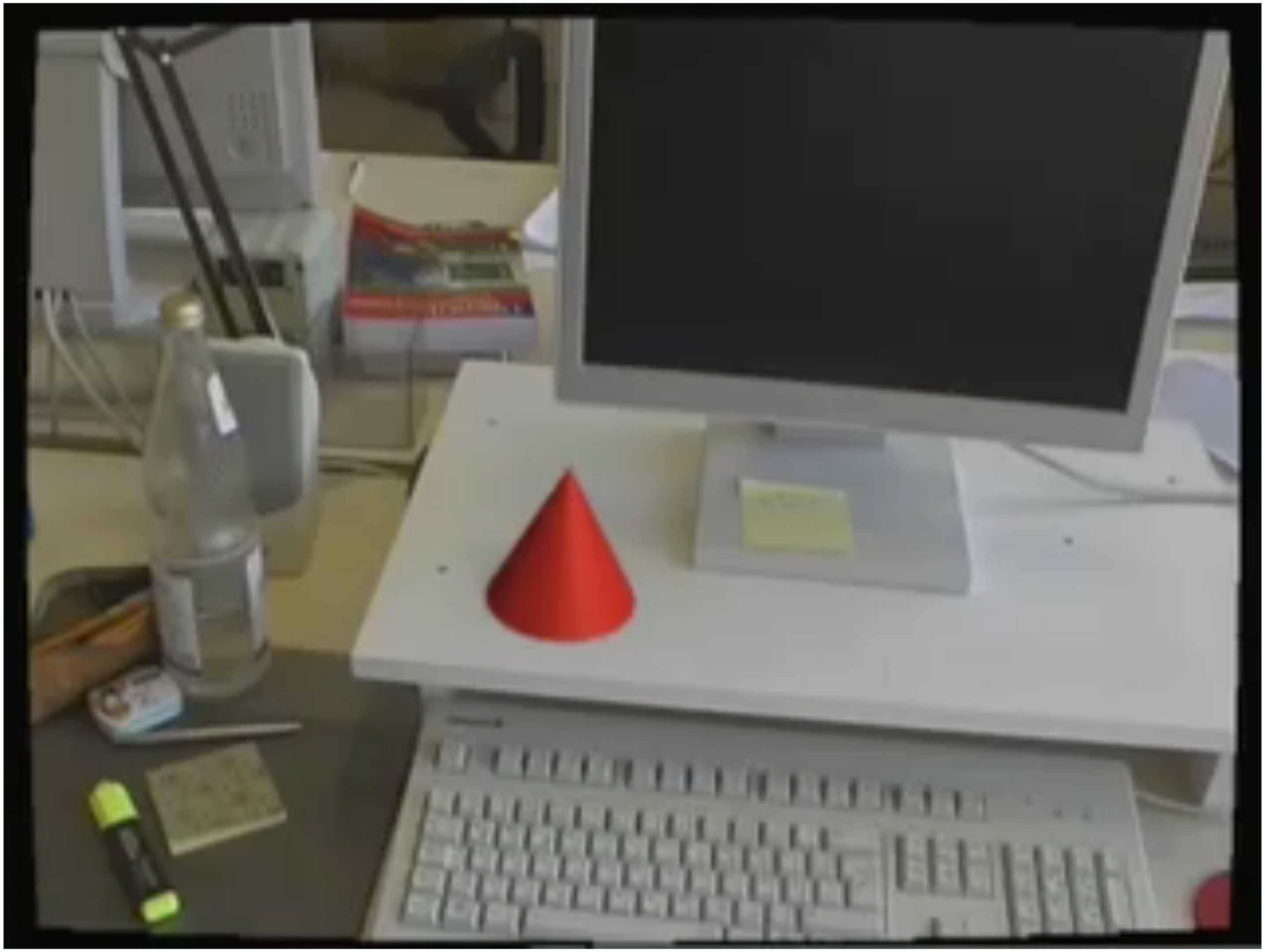


M. Pollefeys, L. Van Gool, M. Vergauwen, F. Verbiest, K. Cornelis, J. Tops, R. Koch, Visual modeling with a hand-held camera, International Journal of Computer Vision 59(3), 207-232, 2004

Rendering and compositing

- Rendering:
  - take camera model, object model, lighting model, make a picture
  - very highly developed and well understood subject
  - many renderers available; tend to take a lot of skill to use (Luxrender)
- Compositing:
  - place two images on top of one another
  - new picture using some pixels from one, some from the other
  - example:
    - green screening
      - take non-green pixels from background, non-bg pixels from top





Recall: Affine Cameras - I

- And this becomes (for the relevant group of points)

$$\begin{pmatrix} U \\ V \\ W \end{pmatrix} = \mathcal{C} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \mathcal{W} \begin{pmatrix} X \\ Y \\ Z \\ T \end{pmatrix}$$

Hide the scale in here

- We will see further simplifications soon

Scaled orthographic cameras

$$\begin{pmatrix} U \\ V \\ W \end{pmatrix} = \mathcal{C} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \mathcal{W} \begin{pmatrix} X \\ Y \\ Z \\ T \end{pmatrix}$$

- Alternatively
  - the camera film plane has
    - two axes, u and v
    - an origin, at (tx, ty)
  - axes are at right angles
  - axes are the same length
  - point in 3D is $(x, y, z) = \mathbf{x}$
  - equation:

$$\mathbf{x} \rightarrow (\mathbf{u} \cdot \mathbf{x} + t_x, \mathbf{v} \cdot \mathbf{x} + t_y)$$

Simplify

- Now place the 3D origin at center of gravity of points
  - ie mean of x over all points is zero, mean of y is zero, mean of z is zero
- Camera origin at center of gravity of image points
  - we see all of them, so we can compute this
  - this is the projection of 3D center of gravity
- Now camera becomes

$$\mathbf{x} \to (\mathbf{u} \cdot \mathbf{x}, \mathbf{v} \cdot \mathbf{x})$$

- Index for points, views

$$\mathbf{x}_j \to (\mathbf{u}_i \cdot \mathbf{x}_j, \mathbf{v}_i \cdot \mathbf{x}_j)$$

Multiple views

- More notation:
  - write $x_{i,j}$ for the first (x) coordinate of the i'th picture of the j'th point
  - write $y_{i,j}$ for the second (y) coordinate of the i'th picture of the j'th point
- We had: $\mathbf{x}_j \rightarrow \left(\mathbf{u}_i \cdot \mathbf{x}_j, \mathbf{v}_i \cdot \mathbf{x}_j\right)$
- Rewrite:

$$\begin{pmatrix} x_{i,j} \\ y_{i,j} \end{pmatrix} = \begin{pmatrix} \mathbf{u}_i^T \\ \mathbf{v}_i^T \end{pmatrix} \mathbf{x}_j$$

Multiple views

$$
\begin{pmatrix}
x_{1,1} & x_{1,2} & \dots & x_{1,n} \\
x_{2,1} & x_{2,2} & \dots & x_{2,n} \\
\dots & & & \\
y_{m,1} & y_{m,2} & \dots & y_{m,n} \\
y_{1,1} & y_{1,2} & \dots & y_{1,n} \\
y_{2,1} & y_{2,2} & \dots & y_{2,n} \\
\dots & & & \\
y_{m,1} & y_{m,2} & \dots & y_{m,n}
\end{pmatrix}
=
\begin{pmatrix}
\mathbf{u}_1^T \\
\mathbf{u}_2^T \\
\dots \\
\mathbf{u}_m^T \\
\mathbf{v}_1^T \\
\mathbf{v}_2^T \\
\dots \\
\mathbf{v}_m^T
\end{pmatrix}
\begin{pmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \dots & \mathbf{x}_n \end{pmatrix}
$$

$$\mathcal{D} = \mathcal{V}\mathcal{X}$$

Data - observed!

Multiple views

- The data matrix has rank 3!
  - so we can factor it into an mx3 factor and a 3xn factor
  - (tall+thin)x(short+fat)
  - so we know what to do; SVD -> factors
    - recall SVD from IRLS!
- These factors are not unique
  - assume A is 3x3 with rank 3, we get symmetry below

$$\mathcal{D} = \mathcal{T}\mathcal{S} = (\mathcal{T}\mathcal{A})(\mathcal{A}^{-1}\mathcal{S})$$

Camera and reconstruction

- Can choose factors uniquely
  - recall v_i, u_i are
    - at right angles
    - same length
- Algorithm
  - form D
  - factor
  - now choose A so that v_i, u_i are at right angles, same length
    - by numerical optimization
- What if there are missing points?
  - Fairly simple optimization trick, following slides

Factoring without all points

- Write D for the data matrix, W for a mask matrix
  - W_ij=0 if that entry of D is unknown, =1 if it is known
- Strategy:
  - choose S, T to minimize

$$\sum_{i,j} W_{ij}(D_{ij} - \sum_{k} T_{ik} S_{kj})^2$$

  - now multiply these S, T - the result is the whole of D
    - i.e. holes are filled in
  - we expect this to work even if D has many holes in it because
    - there are few parameters in S, T

Factors with missing points

- How to minimize? set the gradient to zero
- gradient with respect to T_uv is

$$2\sum_j W_{uj}(D_{uj} - \sum_k T_{uk}S_{kj})S_{vj}$$

- gradient with respect to S_uv is

$$2\sum_i W_{iv}(D_{iv} - \sum_k T_{ik}S_{kv})T_{iu}$$

Software

- Colmap
  - open source SFM at very large scale
  - backbone of many other projects
  - https://demuc.de/colmap/

Notice there are TWO products here

$$\begin{pmatrix} x_{1,1} & x_{1,2} & \dots & x_{1,n} \\ x_{2,1} & x_{2,2} & \dots & x_{2,n} \\ \dots & & & \\ y_{m,1} & y_{m,2} & \dots & y_{m,n} \\ y_{1,1} & y_{1,2} & \dots & y_{1,n} \\ y_{2,1} & y_{2,2} & \dots & y_{2,n} \\ \dots & & & \\ y_{m,1} & y_{m,2} & \dots & y_{m,n} \end{pmatrix} = \begin{pmatrix} \mathbf{u}_1^T \\ \mathbf{u}_2^T \\ \dots \\ \mathbf{u}_m^T \\ \mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \dots \\ \mathbf{v}_m^T \end{pmatrix} \begin{pmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \dots & \mathbf{x}_n \end{pmatrix}$$

Points in 3D

Estimates of camera rotation

What happened to translation?

Key takeaway

- Multiple views of multiple points yields
  - point positions
  - camera rotations
- IF
  - you can match
- We’ll do more detailed versions in various cases
  - but it’s all basically this point