Structure from Motion

Structure from Motion

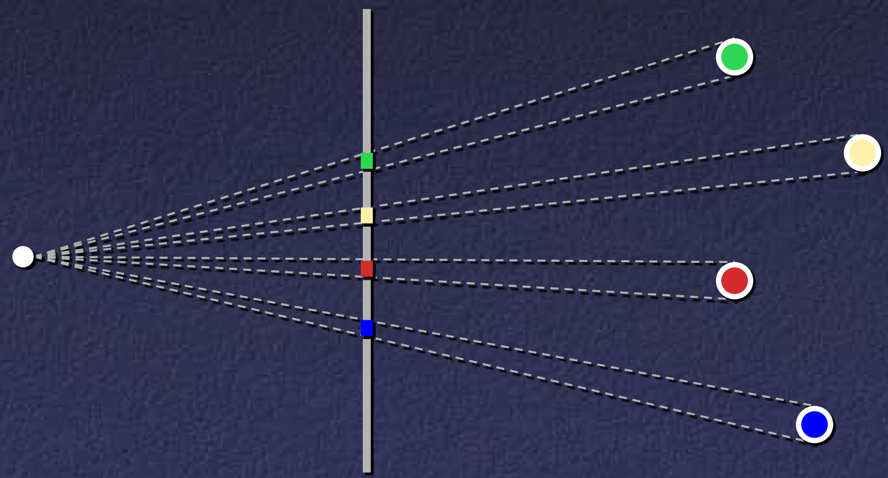
- For now, static scene and moving camera
  - Equivalently, rigidly moving scene and static camera
- Limiting case of stereo with many cameras
- Limiting case of multiview camera calibration with *unknown* target
- Given n points and N camera positions, have $2nN$ equations and $3n+6N$ unknowns

Approaches

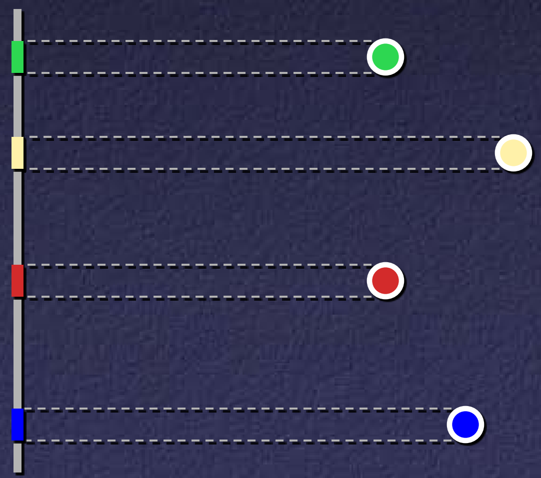
- Obtaining point correspondences
  - Optical flow
  - Stereo methods: correlation, feature matching
- Solving for points and camera motion
  - Nonlinear minimization (bundle adjustment)
  - Various approximations…

Orthographic Approximation

- Simplest SFM case: camera approximated by orthographic projection



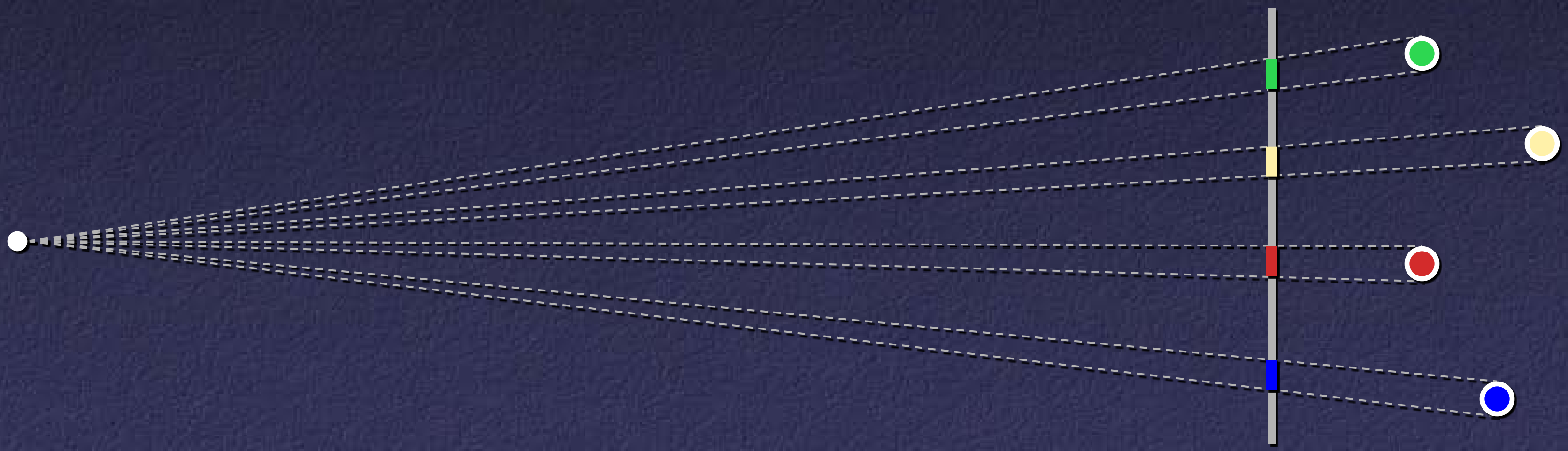
Perspective



Orthographic

Weak Perspective

- An orthographic assumption is sometimes well approximated by a telephoto lens



Weak Perspective

Consequences of Orthographic Projection

- Translation perpendicular to image plane cannot be recovered
- Scene can be recovered up to scale (if weak perspective)

Orthographic Structure from Motion

- Method due to Tomasi & Kanade, 1992
- Assume n points in 3D space $\mathbf{p}_1$ .. $\mathbf{p}_n$
- Observed at N points in time at image coordinates (x_{ij}, y_{ij}), $i = 1..N$, $j=1..n$
  - Feature tracking, optical flow, etc.
  - *All* points visible in *all* frames

Orthographic Structure from Motion

- Write down matrix of data

$$
\begin{array}{ccc}
 & \text{Points} \rightarrow & \\
\mathbf{D} = & \begin{bmatrix}
x_{11} & \cdots & x_{1n} \\
\vdots & \ddots & \vdots \\
x_{N1} & \cdots & x_{Nn} \\
y_{11} & \cdots & y_{1n} \\
\vdots & \ddots & \vdots \\
y_{N1} & \cdots & y_{Nn}
\end{bmatrix} & \begin{array}{c} \text{Frames} \downarrow \\ \\ \\ \text{Frames} \downarrow \end{array}
\end{array}
$$

Orthographic Structure from Motion

- Step 1: find translation
- Translation *perpendicular* to viewing direction cannot be obtained
- Translation *parallel* to viewing direction equals motion of average position of all points

Orthographic Structure from Motion

- After finding translation, subtract it out (i.e., subtract average of each row)

$$\tilde{\mathbf{D}} = \begin{bmatrix} x_{11} - \bar{x}_1 & \cdots & x_{1n} - \bar{x}_1 \\ \vdots & \ddots & \vdots \\ x_{N1} - \bar{x}_N & \cdots & x_{Nn} - \bar{x}_N \\ y_{11} - \bar{y}_1 & \cdots & y_{1n} - \bar{y}_1 \\ \vdots & \ddots & \vdots \\ y_{N1} - \bar{y}_N & \cdots & y_{Nn} - \bar{y}_N \end{bmatrix}$$

Orthographic Structure from Motion

- Step 2: try to find rotation
- Rotation at each frame defines local coordinate axes $\hat{\mathbf{i}}$, $\hat{\mathbf{j}}$, and $\hat{\mathbf{k}}$
- Then $\tilde{x}_{ij} = \hat{\mathbf{i}}_i \cdot \tilde{\mathbf{p}}_j$, $\tilde{y}_{ij} = \hat{\mathbf{j}}_i \cdot \tilde{\mathbf{p}}_j$

Orthographic Structure from Motion

- So, can write $\tilde{\mathbf{D}} = \mathbf{RS}$ where R is a "rotation" matrix and S is a "shape" matrix

$$\tilde{\mathbf{D}} = \begin{bmatrix} x_{11} - \bar{x}_1 & \cdots & x_{1n} - \bar{x}_1 \\ \vdots & \ddots & \vdots \\ x_{N1} - \bar{x}_N & \cdots & x_{Nn} - \bar{x}_N \\ y_{11} - \bar{y}_1 & \cdots & y_{1n} - \bar{y}_1 \\ \vdots & \ddots & \vdots \\ y_{N1} - \bar{y}_N & \cdots & y_{Nn} - \bar{y}_N \end{bmatrix} \qquad \mathbf{R} = \begin{bmatrix} \hat{\mathbf{i}}_1{}^{\mathrm{T}} \\ \vdots \\ \hat{\mathbf{i}}_N{}^{\mathrm{T}} \\ \hat{\mathbf{j}}_1{}^{\mathrm{T}} \\ \vdots \\ \hat{\mathbf{j}}_N{}^{\mathrm{T}} \end{bmatrix} \qquad \mathbf{S} = \begin{bmatrix} \tilde{\mathbf{p}}_1 & \cdots & \tilde{\mathbf{p}}_n \end{bmatrix}$$

Orthographic Structure from Motion

- Goal is to factor $\tilde{\mathbf{D}}$
- Before we do, observe that *rank*($\tilde{\mathbf{D}}$) *should* be 3 (in ideal case with no noise)
- Proof:
  - Rank of **R** is 3 unless no rotation
  - Rank of **S** is 3 iff have noncoplanar points
  - Product of 2 matrices of rank 3 has rank 3
- With noise, *rank*($\tilde{\mathbf{D}}$) might be > 3

SVD

- Goal is to factor $\tilde{\mathbf{D}}$ into $\mathbf{R}$ and $\mathbf{S}$
- Apply SVD: $\tilde{\mathbf{D}} = \mathbf{UWV}^{\mathrm{T}}$
- But $\tilde{\mathbf{D}}$ should have rank 3 $\Rightarrow$ all but 3 of the w_i should be 0
- Extract the top 3 w_i, together with the corresponding columns of $\mathbf{U}$ and $\mathbf{V}$

Factoring for Orthographic Structure from Motion

- After extracting columns, $\mathbf{U}_3$ has dimensions $2N \times 3$ (just what we wanted for $\mathbf{R}$)
- $\mathbf{W}_3 \mathbf{V}_3^\mathrm{T}$ has dimensions $3 \times n$ (just what we wanted for $\mathbf{S}$)
- So, let $\mathbf{R}^* = \mathbf{U}_3$, $\mathbf{S}^* = \mathbf{W}_3 \mathbf{V}_3^\mathrm{T}$

Affine Structure from Motion

- The **i** and **j** entries of $\mathbf{R}^*$ are not, in general, unit length and perpendicular
- We have found motion (and therefore shape) up to an affine transformation
- This is the best we could do if we didn't assume orthographic camera

Ensuring Orthogonality

- Since $\tilde{\mathbf{D}}$ can be factored as $\mathbf{R}^* \mathbf{S}^*$, it can also be factored as $(\mathbf{R}^*\mathbf{Q})(\mathbf{Q}^{-1}\mathbf{S}^*)$, for any $\mathbf{Q}$
- So, search for $\mathbf{Q}$ such that $\mathbf{R} = \mathbf{R}^* \mathbf{Q}$ has the properties we want

Ensuring Orthogonality

- Want $\left(\hat{\mathbf{i}}_i^{*\mathrm{T}}\mathbf{Q}\right)\cdot\left(\hat{\mathbf{i}}_i^{*\mathrm{T}}\mathbf{Q}\right)=1$ or $\hat{\mathbf{i}}_i^{*\mathrm{T}}\mathbf{Q}\mathbf{Q}^{\mathrm{T}}\hat{\mathbf{i}}_i^{*}=1$

$$\hat{\mathbf{j}}_i^{*\mathrm{T}}\mathbf{Q}\mathbf{Q}^{\mathrm{T}}\hat{\mathbf{j}}_i^{*}=1$$

$$\hat{\mathbf{i}}_i^{*\mathrm{T}}\mathbf{Q}\mathbf{Q}^{\mathrm{T}}\hat{\mathbf{j}}_i^{*}=0$$

- Let $\mathbf{T} = \mathbf{Q}\mathbf{Q}^{\mathrm{T}}$
- Equations for elements of **T** – solve by least squares
- Ambiguity – add constraints $\mathbf{Q}^{\mathrm{T}}\hat{\mathbf{i}}_1^{*}=\begin{bmatrix}1\\0\\0\end{bmatrix}, \mathbf{Q}^{\mathrm{T}}\hat{\mathbf{j}}_1^{*}=\begin{bmatrix}0\\1\\0\end{bmatrix}$

Ensuring Orthogonality

- Have found $\mathbf{T} = \mathbf{Q}\mathbf{Q}^T$
- Find $\mathbf{Q}$ by taking "square root" of $\mathbf{T}$
  - Cholesky decomposition if $\mathbf{T}$ is positive definite
  - General algorithms (e.g. **`sqrtm`** in Matlab)

Orthogonal Structure from Motion

- Let's recap:
  - Write down matrix of observations
  - Find translation from avg. position
  - Subtract translation
  - Factor matrix using SVD
  - Write down equations for orthogonalization
  - Solve using least squares, square root
- At end, get matrix $\mathbf{R} = \mathbf{R}^* \mathbf{Q}$ of camera positions and matrix $\mathbf{S} = \mathbf{Q}^{-1}\mathbf{S}^*$ of 3D points

Results

- Image sequence

[Tomasi & Kanade]

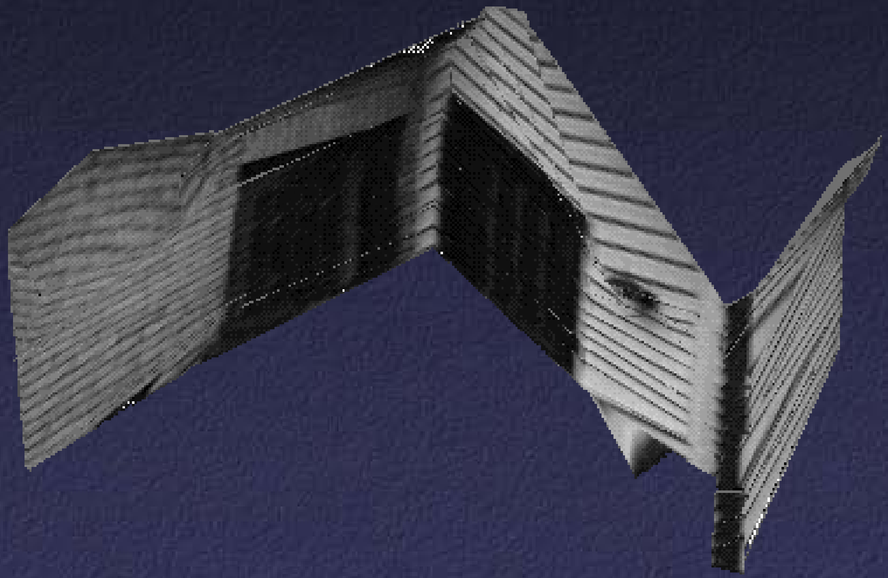
Results

- Tracked features

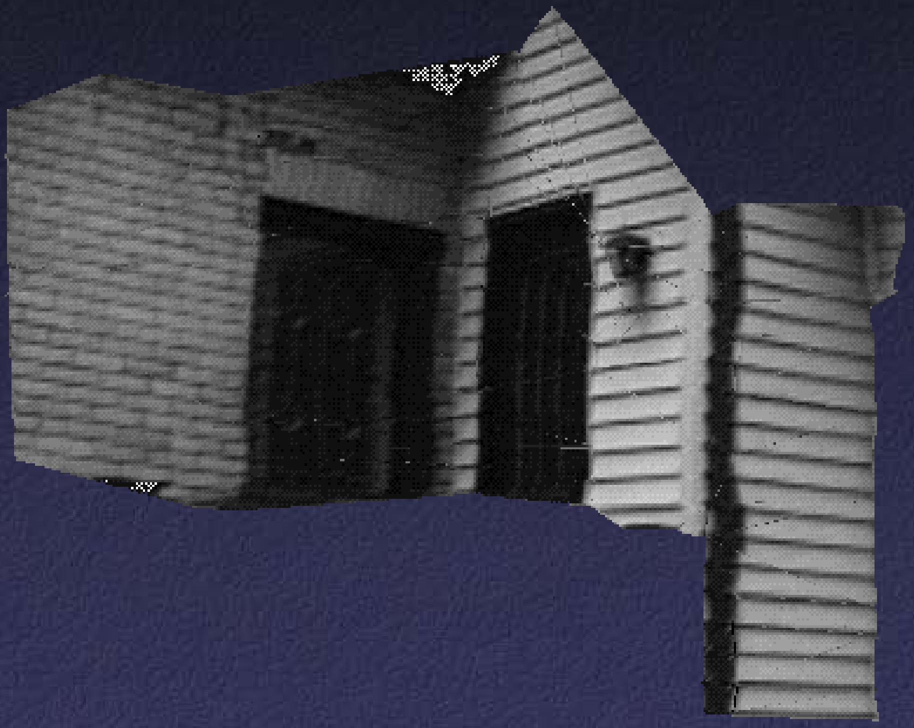
[Tomasi & Kanade]

Results

- Reconstructed shape



Top view



Front view

[Tomasi & Kanade]

Orthographic → Perspective

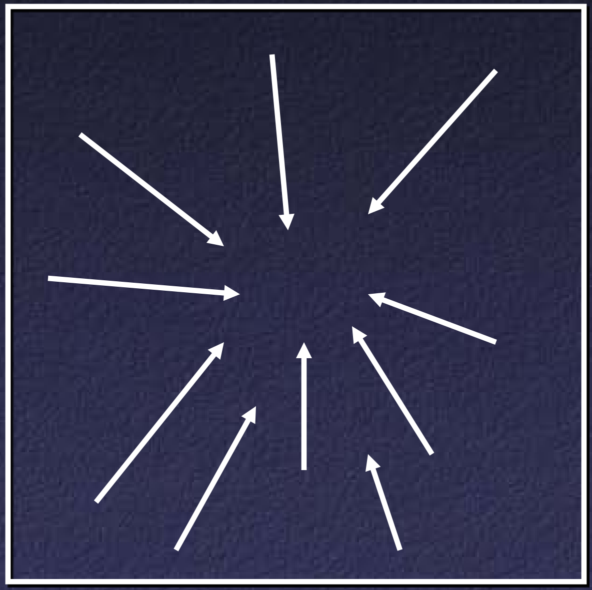
- With orthographic or “weak perspective” can’t recover all information
- With full perspective, can recover more information (translation along optical axis)
- Result: can recover geometry and full motion up to global scale factor

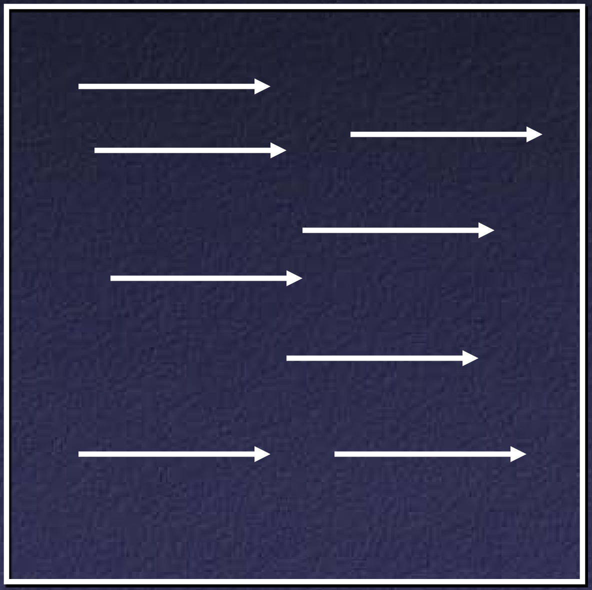
Perspective SFM Methods

- Bundle adjustment (full nonlinear minimization)
- Methods based on factorization
- Methods based on fundamental matrices
- Methods based on vanishing points

Motion Field for Camera Motion

- Translation:

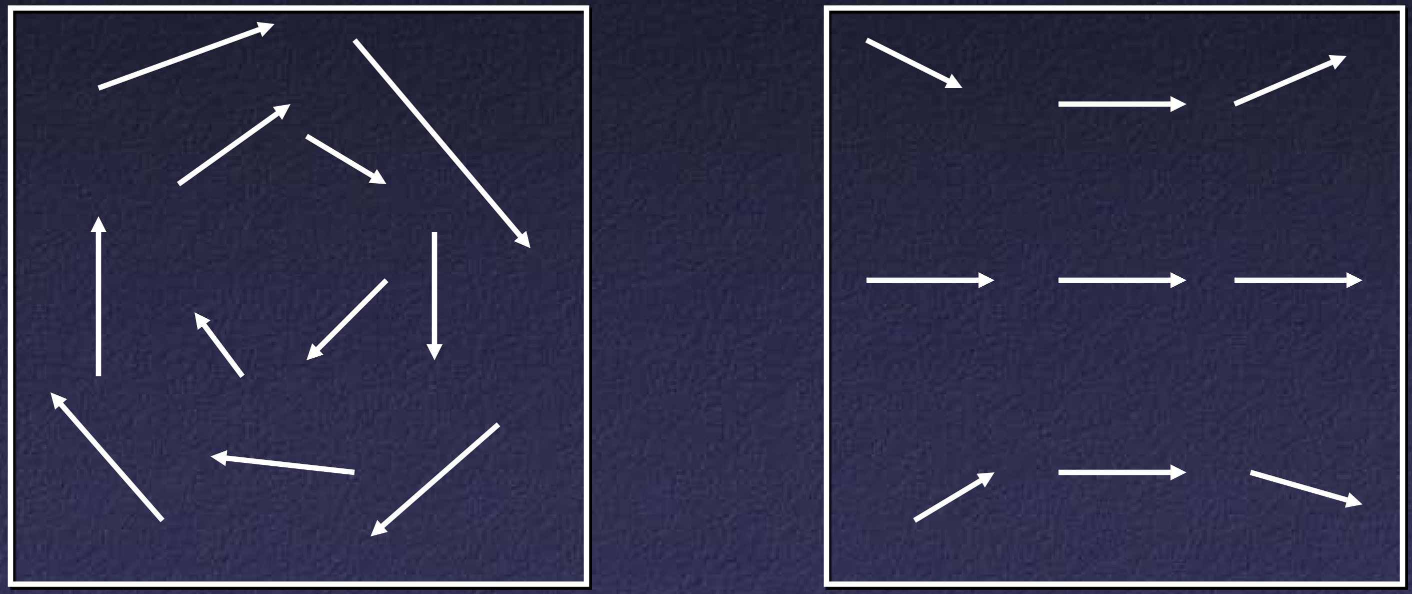




- Motion field lines converge (possibly at ∞)

Motion Field for Camera Motion

- Rotation:



- Motion field lines do not converge

Motion Field for Camera Motion

- Combined rotation and translation: motion field lines have component that converges, and component that does not
- Algorithms can look for vanishing point, then determine component of motion around this point
- "Focus of expansion / contraction"
- "Instantaneous epipole"

Finding Instantaneous Epipole

- Observation: motion field due to translation depends on depth of points
- Motion field due to rotation does not
- Idea: compute *difference* between motion of a point, motion of neighbors
- Differences point towards instantaneous epipole

SVD (Again!)

- Want to fit direction to all Δv (differences in optical flow) within some neighborhood
- PCA on matrix of Δv
- Equivalently, take eigenvector of $\mathbf{A} = \Sigma(\Delta v)(\Delta v)^T$ corresponding to largest eigenvalue
- Gives direction of parallax l_i in that patch, together with estimate of reliability

SFM Algorithm

- Compute optical flow
- Find vanishing point (least squares solution)
- Find direction of translation from epipole
- Find perpendicular component of motion
- Find velocity, axis of rotation
- Find depths of points (up to global scale)