Structure from Motion – Part II

• Given optical flow, recover 3D motion & depth
  – Basic equations
  – Two intuitive but iterative algorithms
  – Closed form algorithm based on parallax

• Appreciate what SFM offers
  – I move, therefore I see

• Appreciate limitations of SFM
  – scale-ambiguity
  – rotation-translation confusion

Structure from Motion – Part II

• Given optical flow, calculate 3D motion and depth
  – Need to relate 3D Motion & depth to 2D optical flow
• Assume a camera moving in a static environment
• Camera motion expressed as a translation and a rotation.
3D Motion of Camera

- \( T \) = the translational component of the camera motion
- \( \omega \) = the rotational velocity
- \( P = \) the position vector \([X \ Y \ Z]^T\)

Relative velocity of \( P \):
\[
V = -T - \omega \times P
\]

Relating 3D Motion to 2D Motion Field

Perspective Projection: \((x,y) = f \frac{(X,Y)}{Z}\)

Taking derivative on both side, we have
\[
x' = f(X'/Z - XZ'/Z^2)
\]
\[
y' = f(Y'/Z - YZ'/Z^2)
\]

On LHS, we have \((\frac{dx}{dt}, \frac{dy}{dt})\) i.e. flow \((v_x, v_y)\)

On RHS, we need \((\frac{dX}{dt}, \frac{dY}{dt}, \frac{dZ}{dt})\) i.e. \((V_x, V_y, V_z)\)
Relating 3D Motion to 2D Motion Field

\[ V = -T - \omega \times P, \]

\[ V_x = -T_x - \omega_y Z + \omega_z Y \]
\[ V_y = -T_y - \omega_z X + \omega_x Z \]
\[ V_z = -T_z - \omega_y Y + \omega_x X \]

Substituting,

\[ v_x = \frac{T_x f - \omega_y f + \omega_x y + \omega_{xy} \frac{x}{f} - \omega_x x^2}{Z} \]
\[ v_y = \frac{T_y f - \omega_z f - \omega_x x - \omega_{xy} \frac{y}{f} + \omega_y y^2}{Z} \]

Note: In the differential case like motion here, \( T \) denotes velocity vector, whereas in the discrete case like stereo, \( T \) is a displacement vector.

• \((v_x, v_y)_{trans}^{trans}\) the transational flow contains information about structure of the scene.
• \((v_x, v_y)_{rot}^{rot}\) the rotational flow is independent of \( Z \).
Pure translation

- When camera motion is only translation, then we have

\[ \omega_x \cdot \omega_y - \omega_z = 0 \]

\[ v_x = \frac{T_x x - T_z f_z}{Z} \]
\[ v_y = \frac{T_x y - T_z f_z}{Z} \]
\[ v_z = \frac{f_T_x}{T_z} \]
\[ v_x = (x - x_0) \frac{T_z}{Z} \]
\[ v_y = (y - y_0) \frac{T_z}{Z} \]

- Consider the special point \((fT_x/T_z, fT_y/T_z)\):
  - This is the “image” of the velocity vector onto the image plane. It is located at where the translation vector cuts the image plane.
  - The motion at this point must be 0 since the surface point along this ray stays on the ray as the camera moves (our equations evaluate to 0 at this point too)

\[ \omega \cdot \omega = \omega_z \]

\[ \frac{v_x}{v_z} = \frac{x - x_0}{y - y_0} \frac{T_z}{Z} \]

- In stereo context, this point is known as \( \text{epipole} \).
Pure translation

- \( T = [0, 0, 1] \); FOE \((x_0, y_0)\) ?
- \( T = [1, 0, 0] \); FOE \((x_0, y_0)\) ?

Where is the FOE given that the 2 flows are purely translational?

Scale ambiguity in \( T \)

- So if we have optical flow, we can calculate the direction of translation in the form of FOE. But can we recover the absolute magnitude of the 3 components \( T_X, T_Y, T_Z \)?
- No, we can only recover \( T \) up to a scale ambiguity. This ambiguity is clear from \((T_X, T_Y\) occur in ratio with \( T_Z \).

\[
\begin{align*}
  x' &= \frac{fT_X}{T_Z} \\
  y' &= \frac{fT_Y}{T_Z}
\end{align*}
\]

Error in textbook: P185: 5th from bottom: it should be “if \( T_Z > 0 \)”, not \( T_Z < 0 \), and 4th line from bottom, it should be “if \( T_Z < 0 \)”, not \( T_Z > 0 \). P186; 3rd line from bottom: it should be “also proportional”, not “also inversely proportional”.
Scale Ambiguity in Z

- There is also scale ambiguity in Z
  - \( T_x, T_y, T_z \) occur in ratio with \( Z \).
    \[
    v_x = \frac{T_x x - T_x f}{Z}
    \]
    \[
    v_y = \frac{T_y y - T_y f}{Z}
    \]
- Same optic flow field generated by two similar surfaces undergoing similar motions: \((T_x, T_y, T_z, Z)\) and \((k T_x, k T_y, k T_z, k Z)\).
- If we have computed the FOE of an image sequence then we can compute the (scaled) depth to visible points in the scene
  \[
  v_x = (x - x_0) \frac{T_x}{Z} \quad \Rightarrow \quad \frac{Z}{T_z} = \frac{x - x_0}{v_x}
  \]
  - Since all depths in the scene can only be recovered up to a common scale factor, we sometimes just use \( Z / T_z \) (depth scaled by \( T_z \)) as the solution for \( Z \).

General 3D Motion (SFM)

- So far, we have considered the simple case of pure translation.
  - To solve general 3D motion (with Rotation \( R \) and translation \( T \)) is a difficult problem!
  - One key problem is the coupling between \( R \) & \( T \). Small rotations about the y (x) axis are easy to confuse with translations in x (y).

Small rotations about the y axis are easy to confuse with translations in x.

Pure translation along Z-axis.

Can be mistaken as a pure translation heading in a direction slightly off the Z-axis.

NB: FOE is still at the cross.
List pair of motions that are hard to separate:

General 3D Motion (SFM) – More practical problems in solving SFM

-- Computing optical flow is difficult.
- Optical flow algorithms need to integrate information over small image neighborhoods, assuming smoothness of flow.
- If those neighborhoods overlap a boundary between an object and the background, smoothness assumptions are violated and the result will be wrong.

-- Independently moving objects confuse 3D motion estimation algorithms
- their motion is inconsistent with the rigid camera motion
- Motion field of the moving object is inconsistent with the radial motion field emanating from FOE
Structure from Motion

- What happens if you can’t recover the 3D motion perfectly
  - The structure that you perceive will be distorted

Solving General SFM

\[
\begin{align*}
\mathbf{v}_x &= \left(x - x_0\right) \frac{T}{Z} - \omega_z f + \omega_y y + \omega_x x + \omega_{xy} y + \omega_{xz} z + \omega_{yz} z
\end{align*}
\]

\[
\begin{align*}
\mathbf{v}_y &= \left(y - y_0\right) \frac{T}{Z} + \omega_z f - \omega_x x - \omega_{xy} y - \omega_{xz} z - \omega_{yz} z
\end{align*}
\]

- For N image points, there are 2N equations (each point provides 2 optical flow equation) with N+5 unknowns (N depths, 2 for FOE, 3 for rotation).
  - Possible to solve with numerical method but dimension too high.

- Usual method: factor out Z from the 2 equations

  \[
  (v_x - v_x^{\text{Rot}}, v_y - v_y^{\text{Rot}}) \cdot (y - y_0, -(x - x_0)) = 0
  \]
  - N image points; N equations; 5 unknowns \(x_0, y_0, \omega_x, \omega_y, \omega_z,\)

- Essentially, given optical flow, algorithms try to find a set of \(x_0, y_0, \omega_x, \omega_y, \omega_z,\) which can minimize

\[
\sum_{i=1}^{N} \sum_{j=1}^{N} (v_{x_i} - v_{x_j}^{\text{Rot}})(y - y_0) - (v_{y_i} - v_{y_j}^{\text{Rot}})(x - x_0) = 0
\]


Simple SFM Algorithm

- Still a five dimensional search. Can further decompose the parameters to reduce the search dimension.
  - 2D search for FOE, obtain rotation in closed form from FOE.
  - 3D search for rotation, obtain FOE in closed form from rotation.
- One way is to first search for the translational parameters (FOE).
  - Each hypothesized FOE defines a set of emanating lines
  - Project optical flow in the direction \( \perp \) to these lines.
  - It would only contain rotational flow if the FOE is chosen correctly.
  - Fit the 3 rotational parameters (e.g. LS) & obtain solution in closed form. Ex: write down the LS equation.
  - check the residual for goodness of fit.

- Another way is to first search for the 3 rotational parameters. (e.g. Prazdny 80)
  - Given candidate rotation, can remove rotational flow completely
    \[
    \begin{align*}
    (v_x)_{est} &= -\omega_y f + \omega_z y + \frac{\omega_y x y}{f} - \frac{\omega_z x^2}{f} \\
    (v_y)_{est} &= \omega_x f - \omega_z x - \frac{\omega_y x y}{f} + \frac{\omega_z y^2}{f}
    \end{align*}
    \]
  - If the rotational parameters are chosen correctly, then after “de-rotation”, all flow field should meet at FOE. Why?
  - Check the intersections of the de-rotated flow and choose rotation such that the dispersion of the intersections is smallest.
- E.g. Given that the rotation is given by (0, 0, 0.1), and the optical flows at the feature points (1,0) and (1,1) is given by (1, -0.1) and (1.1, 0.9) respectively, find the FOE \((x_0, y_0)\).
- The above methods are conceptually simple, but their solutions require iteration which is time consuming.
Motion Parallax

- Motion parallax: Consider two visual features at different depths whose projections on the image plane are coincident, their relative motion field – motion parallax -- does not depend on the rotational component of motion in 3-D space.
- Relative motion (ie difference) between the 2 flow fields:
  \[
  \Delta v_x = v_{x,2}^{trans} - v_{x,1}^{trans} = (T_2 x - T_1 f)(\frac{1}{Z_2} - \frac{1}{Z_1})
  \]
  \[
  \Delta v_y = v_{y,2}^{trans} - v_{y,1}^{trans} = (T_2 y - T_1 f)(\frac{1}{Z_2} - \frac{1}{Z_1}).
  \]
  \[
  \frac{\Delta v_y}{\Delta v_x} = \frac{y - y_0}{x - x_0}
  \]
  direction of motion parallax
  \[
  \Delta v_{x1}, \Delta v_{y1}
  \]
  \[
  \Delta v_{x2}, \Delta v_{y2}
  \]
- FOE can be determined.

- Problem: not many pairs of points would exactly satisfy the coincidence condition.
- Approximate motion parallax: regard the flow difference between 2 nearby points as noisy estimate of the true motion parallax.
Approximate Motion Parallax Algorithm

- Obtain approximate motion parallax
- Compute FOE from approximate motion parallax
- Compute rotation & Z from FOE

- At each neighborhood, solve $\text{LS } Ax = 0$ using SVD, where $x$ is a unit vector $\perp$ to the parallax $\begin{bmatrix} \Delta v_x \\ \Delta v_y \end{bmatrix}$
- Solve $\text{LS } A' \begin{bmatrix} x_v \\ y_v \end{bmatrix} = b$ using SVD
- Solve $\text{LS } A'' \begin{bmatrix} \omega_x \\ \omega_y \end{bmatrix} = b'$ using SVD

Motion Parallax

- To find best estimate of parallax ($\Delta v_x, \Delta v_y$) using various noisy estimates ($\Delta v_x', \Delta v_y'$). Determine the eigenvalues & eigenvectors of the matrix:
  $\begin{bmatrix} \sum \Delta^2 v_x & \sum \Delta v_x \Delta v_y \\ \sum \Delta v_x \Delta v_y & \sum \Delta^2 v_y \end{bmatrix}$

- The eigenvector associated with the greater eigenvalue is the best estimate of the motion parallax within the patch.
  - If rank is 1, data can be fitted perfectly and is reliable. (in other words, degree of freedom/ number of basis / number of principal component is 1)
  - Can use the ratio of the two eigenvalues as a measure of the estimate’s reliability.
Interlude: Linear minimization

- Consider the modified problem: finding the direction of $x$ that is most perpendicular to all the parallax

  \[
  \begin{align*}
  &\Delta v_x \\
  &\Delta v_y \\
  &x
  \end{align*}
  \]

- Form the matrix $A$, where $i^{th}$ row is given by $(\Delta v_{xi}, \Delta v_{yi})$.
- Amounts to solving $Ax=0$, for non-zero $x$.

- Choose $x$ to be the eigenvector associated with the smallest eigenvalue of $A^TA$. Recall the same result from the section on SVD. Why is this?
- $x$ can only be determined up to a scale, so, choose $x$ to be a unit vector, $||x||=1$.
- We want to find $x$ s.t. $\varepsilon=Ax$ is minimum and $||x||=1$. Lagrange multipliers!
- Define cost $C=||\varepsilon||^2 + \lambda (1-||x||^2)$
- Can be rewritten as $C=x^TA^TAx + \lambda (1-x^Tx)$
- Find critical points of $C$, ie, where derivative $dC/dx=0$
Interlude: Linear minimization

• \( \frac{dC}{dx} = 2 A^T Ax - 2 \lambda x = 0 \)
  \[ \Rightarrow A^T Ax = \lambda x \]
  
• This is the eigen equation!
• Any eigenvector of \( A^T A \) is a solution.
• Choose the eigenvector \( e_n \) that minimizes \( || \varepsilon ||^2 \)
  \[
  || \varepsilon ||^2 = (e_n^T A^T)(A e_n) = e_n^T (A^T A e_n)
  = e_n^T e_n \lambda_n = \lambda_n
  \]

Interlude: Linear minimization

• This is minimized by choosing \( x = e_n \) where \( e_n \) is the eigenvector associated with the smallest eigenvalue \( \lambda_n \).

• Our original problem is to find a direction that is most consistent with the direction of the \( n \) lines obtained, ie, we want to maximize \( Ax \).

• So to maximize \( || \varepsilon ||^2 \), choose \( x = e_m \) where \( e_m \) is the eigenvector associated with the largest eigenvalue \( \lambda_m \).
Interlude: Linear minimization

- How to set up A, given n measurements $(\Delta v_x, \Delta v_y)_i, i = 1, 2, \ldots n$? A consists of n rows, with $i^{th}$ row given by $(\Delta v_x, \Delta v_y)$.
- We are trying to find a normal $x = (a, b)$ that are perpendicular to these directions. Thus each equation is of the form $a \Delta v_x + b \Delta v_y = 0$. (the parallax is in the direction $(b, -a)$.
- This normal $x = (a, b)$ is the eigenvector associated with the smallest eigenvalue of $A^T A$; the parallax $(b, -a)$ is the eigenvector associated with the largest eigenvalue of $A^T A$.
- Can solve by eigenvector technique or by solving SVD(A). Columns of V are eigenvectors of $A^T A$.

$$A^T A = \begin{bmatrix} \sum \Delta^2 v_x & \sum \Delta v_x \Delta v_y \\ \sum \Delta v_x \Delta v_y & \sum \Delta^2 v_y \end{bmatrix}$$

Approximate Motion Parallax Algorithm

- Obtain approximate motion parallax
  - Compute FOE from approximate motion parallax
    - At each neighborhood, solve $LS A x = 0$ using SVD, where $x$ is a unit vector $\perp$ to the parallax $\begin{bmatrix} \Delta v_x \\ \Delta v_y \end{bmatrix}$
    - Solve $LS A^T \begin{bmatrix} x_o \\ y_o \end{bmatrix} = b$ using SVD
  - Compute rotation & Z from FOE
    - Solve $LS A^T \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = b'$ using SVD
Motion Parallax

- With several motion parallax computed from N patches, the intersection yields the FOE.

From Block $B_i$ centered at $(x_i, y_i)$, we have used SVD($A_i$) to obtain the parallax direction $(b_i, -a_i)$ and consistency measure $w_i$.

- Each block $B_i$ yields an equation $(a_i, b_i) \cdot (x_i - x_0, y_i - y_0)^T = 0$ (the normal to the parallax must be ⊥ to the emanating lines from FOE).
- Collect equations from all blocks and solve for $(x_0, y_0)$ by LS. Ex. Write down the LS equation.
- Better: use weighted least square. Weight reflects consistency in motion parallax measurement. Each row is weighted (multiplied) by $w_i$ as weight. $w_i$ can be the ratio of the two eigenvalues.

Approximate Motion Parallax Algorithm

- At each neighborhood, solve LS $Ax = 0$ using SVD, where $x$ is a unit vector ⊥ to the parallax

\[
\begin{bmatrix}
\Delta v_x \\
\Delta v_y
\end{bmatrix}
\]

- Solve LS $A' \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} = b$ using SVD

- Solve LS $A'' \begin{bmatrix} \omega_x \\ \omega_y \\ \omega_z \end{bmatrix} = b'$ using SVD

The rest of the problem (rotation & Z) is easy. Refer to the intuitive algorithm for the LS equation.
End of Motion Analysis

• Key points:
  □ Motion equations relating optical flow & 3D motion & Z
  □ Properties of these equations; e.g. scale ambiguity, ambiguity between Rotation & Translation, etc.
  □ Given rotation, how to solve FOE, and vice versa
  □ Parallax Algorithm

• Follow up Activities: (*: Optional)
  □ Revise lecture notes; attempt tutorial.
  □ *Additional / self reading: book & classic papers
    ■ J. Weng etc. Motion and Structure from Image Sequences, 1993.
  □ *Read up Richard Dawkins’ book “Climbing Mount Improbable” for next week.

End of Motion Analysis

• Relevant textbook sections: Trucco (8.1 - 8.3, 8.4.1, 8.5.2)
  • Sect 8.5.1 is not examinable but it describes a technique typical of most SFM methods used in our field: mathematically involved and fraught with limitations.

  The following few slides introduce you to key arguments researchers are engaged in at the cutting edge of this field. They are not examinable but I hope they will give you a broader perspective and further fascinate you and maybe you will take up research in this field.
Is reconstruction the right approach?

• So far, vision has been conceived as a problem of creating hierarchical representations.
  – 2-D images -> primal sketch -> 2½-D sketch -> object-centered descriptions. Known as “from pixels to predicates”

• Vision is described as the process of creating a complete and accurate representation of the scene.

• Thus, much of the motion analysis research has focused on SFM (complete scene recovery), as well as estimating the 3-D motion parameters.

Is reconstruction the right approach?

• But complete scene reconstruction results in:
  – more information than is necessary
  – mathematical difficulty, ill-posedness
  – prolonged time needed to solve motion related problems.

• low-level animals, such as anthropods, insects, and mollusks are still able to solve motion analysis problems
  – even they do not possess powerful computational mechanism to perform 3-D scene reconstruction. E.g.

• One fundamental flaw - the study of the visual system is undertaken in isolation from its environment.
  – Given infinite resources, every problem can be solved in principle but resources are finite
  – vision is always purposeful
Is reconstruction the right approach?

- Agent is always engaged in some tasks, subserved by vision
  - Emerging paradigm of purposive vision
- Possible to divide a visual problem into several sub-tasks and solve them without scene reconstruction
- For example, the task of detecting obstacle
  - Not necessary to compute the exact motion
  - But only to recognize certain patterns of flow evolve in a way that signifies collision.
- Instead of reconstructing the world, recognize entities that are directly relevant to task at hand.
  - Does there always exists an appropriate representation to allows us to directly derive the necessary parameters?

Eyes in biological world

- Must it be camera-type eye?
- Eyes in nature have evolved no fewer than 40 times independently in diverse parts of animal kingdom.
- Eyes “landscape” show 9 basic types of eyes.
Eyes in biological world

- Why flying animals (insects, birds) have panoramic vision?
  - either as compound eye or having camera-type eyes on opposite sides of head
- Deeper mathematical reasons for having panoramic vision?
  - Resolve the confounding between translation and rotation
  - Insect eyes are not just panaromic! It is built from large collection of ommatidia that can be considered as individual cameras.
    - A large collection of stereo systems?

Non-conventional camera systems
Bio-robotics

- In face of errors in 3-D motion estimates, what motion strategy to adopt?
  - Examples in nature: mantis, locust, wasp