Supplement: A Practical Stereo Depth System for Smart Glasses

Jialiang Wang[1*] Daniel Scharstein[1] Akash Bapat[1] Kevin Blackburn-Matzen[2†]
Matthew Yu[1] Jonathan Lehman[1] Suhib Alsisan[1] Yanghan Wang[1] Sam Tsai[1]
Jan-Michael Frahm[1] Zijian He[1] Peter Vajda[1] Michael F. Cohen[1] Matt Uyttendaele[1]

[1]Meta Platforms Inc. [2]Adobe

0.1. Derivation of projection model

Below is an expanded version of Section 4.1 with the full derivation of the projection model.

A 3D point $P = (X, Y, Z)$ projects to pixel coordinates (u_0, v_0), (u_1, v_1) in the two images. Assuming that radial distortion has been corrected for and that intrinsics (focal lengths f_i and principal points c_{x_i}, c_{y_i}) are known, we convert to normalized image coordinates $x_i = (u_i - c_{x_i})/f_i$, $y_i = (v_i - c_{y_i})/f_i$.

Under the above assumptions the cameras are located at $\mathbf{t}_0 = [0\ 0\ 0]^T$ and $\mathbf{t}_1 = [1\ 0\ 0]^T$, and their rotations are $\mathbf{R}_0 = \mathbf{R}(\boldsymbol{\omega}_0)$ and $\mathbf{R}_1 = \mathbf{R}(\boldsymbol{\omega}_1)$. We use a linear approximation for the rotations since we expect the rotational corrections to be small:

$$\mathbf{R}(\boldsymbol{\omega}) \approx \mathbf{I} + [\boldsymbol{\omega}]_\times = \begin{bmatrix} 1 & -\omega_z & \omega_y \\ \omega_z & 1 & -\omega_x \\ -\omega_y & \omega_x & 1 \end{bmatrix}. \quad (1)$$

In normalized image coordinates, point P projects into the left camera at

$$\begin{bmatrix} x_0 \\ y_0 \\ 1 \end{bmatrix} \sim \begin{bmatrix} \mathbf{I} + [\boldsymbol{\omega}_0]_\times \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}. \quad (2)$$

Parametrizing by inverse depth (*disparity*) $d = 1/Z$ we can "unproject" the point

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \sim \begin{bmatrix} X/Z \\ Y/Z \\ 1 \\ 1/Z \end{bmatrix} \sim \left[\begin{array}{c|c} \mathbf{I} - [\boldsymbol{\omega}_0]_\times & \mathbf{0} \\ \hline \mathbf{0} & 1 \end{array}\right] \begin{bmatrix} x_0 \\ y_0 \\ 1 \\ d \end{bmatrix}. \quad (3)$$

Projecting it into the right camera, we get

$$\begin{aligned}
\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} &\sim \left[\begin{array}{c|c} \mathbf{I}+[\boldsymbol{\omega}_1]_\times & \mathbf{0} \end{array}\right] \left[\begin{array}{c|c} \mathbf{I} & \begin{matrix} 1 \\ 0 \\ 0 \end{matrix} \\ \hline \mathbf{0} & 1 \end{array}\right] \left[\begin{array}{c|c} \mathbf{I}-[\boldsymbol{\omega}_0]_\times & \mathbf{0} \\ \hline \mathbf{0} & 1 \end{array}\right] \begin{bmatrix} x_0 \\ y_0 \\ 1 \\ d \end{bmatrix} && (4) \\
&\approx \begin{bmatrix} \mathbf{I}+[\boldsymbol{\omega}_1-\boldsymbol{\omega}_0]_\times \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ 1 \end{bmatrix} + \begin{bmatrix} \mathbf{I}+[\boldsymbol{\omega}_1]_\times \end{bmatrix} \begin{bmatrix} d \\ 0 \\ 0 \end{bmatrix} && (5) \\
&= \begin{bmatrix} \mathbf{I}+[\Delta\boldsymbol{\omega}]_\times \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ 1 \end{bmatrix} + d \begin{bmatrix} 1 \\ -\omega_{z_1} \\ \omega_{y_1} \end{bmatrix}, && (6)
\end{aligned}$$

where $\Delta\boldsymbol{\omega} = \boldsymbol{\omega}_1 - \boldsymbol{\omega}_0$ is the relative orientation of the two cameras. We can see that for $d = 0$ (i.e., a point at infinity), we can only recover the relative orientation $\Delta\boldsymbol{\omega}$. For closer points ($d < 0$) we also get a constraint for absolute roll and absolute pan, but not absolute pitch, as discussed earlier.

If we use $\Delta x = x_1 - x_0$ as our estimate for d and also introduce a scale correction $(1 + \Delta f)$, we have

$$\begin{aligned}
\begin{bmatrix} x_1 \\ y_1 \\ 1+\Delta f \end{bmatrix} &\sim \begin{bmatrix} 1 & \Delta\omega_z & -\Delta\omega_y \\ -\Delta\omega_z & 1 & \Delta\omega_x \\ \Delta\omega_y & -\Delta\omega_x & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ y_0 \\ 1 \end{bmatrix} + \Delta x \begin{bmatrix} 1 \\ -\omega_{z_1} \\ \omega_{y_1} \end{bmatrix} && (7) \\
&= \begin{bmatrix} x_0 + \Delta\omega_z y_0 - \Delta\omega_y + \Delta x \\ y_0 - \Delta\omega_z x_0 + \Delta\omega_x - \omega_{z_1}\Delta x \\ 1 + \Delta\omega_y x_0 - \Delta\omega_x y_0 + \omega_{y_1}\Delta x \end{bmatrix}. && (8)
\end{aligned}$$

Cross-multiplying and dropping higher-order terms we get two equations in the 6 unknowns, where $\Delta y = y_1 - y_0$:

$$\begin{bmatrix} -y_0 x_1 & 1+x_0 x_1 & -y_0 & \Delta x x_1 & 0 & -x_1 \\ 1+y_0 y_1 & -x_0 y_1 & -x_0 & -\Delta x y_1 & -\Delta x & y_0 \end{bmatrix} \begin{bmatrix} \Delta\omega_x \\ \Delta\omega_y \\ \Delta\omega_z \\ \omega_{y_1} \\ \omega_{z_1} \\ \Delta f \end{bmatrix} = \begin{bmatrix} 0 \\ \Delta y \end{bmatrix}. \quad (9)$$

Of these, only the second equation gives us a constraint relating y_0 and y_1. We can collect these equations, one for each matched feature point, and solve the over-constrained system using robust least squares.

*Email: jialiangw@meta.com

†work was done at Meta