Jump: Virtual Reality Video Supplement

Robert Anderson David Gallup Jonathan T. Barron Janne Kontkanen
Noah Snavely Carlos Hernández Sameer Agarwal Steven M. Seitz

Google Inc.

1 Coarse Tile Flow Upsampling

In the main paper we described a technique for producing a coarse per-tile alignment between a pair of images, in which a brute-force normalized SSD computation is used to produce a set of horizontal and vertical displacements and a corresponding confidence of that displacement. That is, for the set of non-overlapping 32×32 tiles in the image of interest we have $\{U_i, V_i, C_i\}$, where U_i, V_i, and C_i are the horizontal displacement, vertical displacement, and confidence, respectively. From this per-tile flow/confidence field we will produce a per-pixel flow/confidence field. To do this, we will apply a series of heuristic operations which lower the confidence of tiles likely to have incorrect flow estimates. From this refined per-tile alignment we produce an upsampled per-pixel flow/confidence field $\{\hat{u}_i, \hat{v}_i, \hat{c}_i\}$ via an adaptive upsampling process which attempts to best warp the tile flow to the structure of the reference image. Each image pair's "forward" per-pixel flow/confidence field is then combined with that pair's corresponding "backward" per-pixel flow/confidence field to model our assumption that the flow field should be symmetric. Our resulting flow/confidence field is fed into the bilateral solver as described in the main paper, which causes the flow to be denoised and inpainted in low-confidence regions but preserved in high-confidence regions. The bilateral solver is an aggressive smoothing operator which performs global optimization across entire video sequences. It is very effective at inpainting low-confidence regions but is not robust to incorrect flow estimates with high confidence. Our coarse flow refinement and upsampling procedure is therefore designed to be very conservative when assigning high confidence to pixels. A small number of very large confidence pixels which reliably indicate motion, are sufficient to inpaint large regions of the image in an edge-aware fashion.

1.1 Repeated Texture

Recall that each tile's motion was estimated by taking the (subpixel) argmin of an SSD image:

$$(U_i, V_i) \approx \arg\min_{u,v} D_i(u, v). \tag{1}$$

Simplifying the structure of $D_i(u, v)$ down to a single point ignores a great deal of information that may be present in $D_i(u, v)$. For example, if there is repeated texture in the other image, then there may be many local minima in $D_i(u, v)$ which have nearly as small a SSD as the global minimum. If this is the case we would ideally propagate all of these minima, however since the filtering stage requires a single estimate for displacement we instead reduce the confidence for this tile. To this end, after extracting the global minimum from $D_i(u, v)$ we extract a second minimum which is at least 32 pixels away from (U_i, V_i) in u and v. Let $d_i = \min_{u,v}(D_i(u, v))$ be the global minimum of D_i corresponding to (U_i, V_i), and let d'_i be the value of D_i at this second minimum. The tile's confidence C_i is updated based on the ratio between d_i and d'_i as follows.

$$C_i \leftarrow C_i \exp\left(-w_r \min(1, \max(0, \frac{(\max(\epsilon_d, d_i) - d'_i r_0)^2}{{d'_i}^2 (r_1 - r_0)^2}))\right), \tag{2}$$

where w_r is a weight that controls the overall effect of the term, ϵ_d is a small value that ensures that this term still has an effect even as d approaches zero and r_0 and r_1 give the range of ratios over which this term transitions from having no effect to having full effect. We use $w_r = 100$, $\epsilon_d = 50$, $r_0 = 0.6$ and r_1=0.8.

1.2 Low Variance Tiles

If the tile of the reference image being matched has very little image texture, then the motion estimated for that tile should be assigned a low confidence. If we were to simply compute the SSD between non-normalized image tiles, using the determinant of the $\mathbf{A}$ matrix in the confidence would naturally encourage this property. But because our images are pre-normalized to have a mean of zero and a standard deviation of 1, our SSD measure assumes all tiles are comparably textured. To this end, for each tile we look at the non-normalized tile and compute it's variance $\mathrm{var}(T)$ then update the tile confidence using

$$C_i \leftarrow C_i \exp\left(-\max\left(0, \frac{w_v}{\mathrm{var}(T)} - \epsilon_v\right)\right). \tag{3}$$

w_v is a weight which controls the strength of this term and ϵ_v is a threshold on variance below which we consider a tile to have low variance. We use $w_v = 100$ and $\epsilon_v = 25$. When calculating $\mathrm{var}(T)$ pixel values in the input image range from 0 to 255.

1.3 Outlier Tiles

We observe that accurate tile flow estimates tend to have nearby tiles with similar flow. Therefore, if none of a tile's neighbors have a flow which is sufficiently close to that tile's flow, we reduce that tile's confidence:

$$C_i \leftarrow C_i \exp\left(-\min_{j \in \mathrm{neigh}(i)} \left(\frac{(U_i - U_j)^2}{\sigma_u^2} + \frac{(V_i - V_j)^2}{\sigma_v^2}\right)\right) \tag{4}$$

Where $\mathrm{neigh}(i)$ are the 4-connected neighbors of tile i and σ_u and σ_v control the scale of the expected variation between a tile's flow and its neighbor. We set $\sigma_u = 16$ and $\sigma_v = 1$, thereby allowing for large neighboring variation in horizontal motion between neighbors (ie, large depth discontinuities) while discouraging vertical motion. By taking the min over each neighbor, a tile's confidence can remain high provided there is at least one neighbor with a similar flow.

1.4 Image Aware Upsampling

Mapping a per-tile flow/confidence field to a per-pixel field requires an upsampling step. Straightforward choices for this upsampling operation can have a large negative impact on the quality of the output. For example, using nearest-neighbor upsampling produces a blocky flow field, which also does not respect the structure of the reference image. Using bilinear or bicubic upsampling often produces egregious oversmoothing artifacts, as interpolation incorrectly assumes that a pixel between four tiles has a motion which is some average of those four tile's motions, when the pixel's motion is likely best modeled as being similar to one or more tiles but not similar to the average of all tiles. We therefore use a modified nearest-neighbor upsampling procedure: we look at the motions of the four tiles which "bound" each pixel and assign each pixel the motion which minimizes the error between a 3×3 window centered on that pixel and the corresponding window in the alternate image indicated by that tile's motion.

$$t_i = \underset{t \in \mathrm{bound}(x,y)}{\arg\min} \sum_{a=-1}^{1} \sum_{b=-1}^{1} \left| I_0' \left(x+a, y+b\right) - I_1' \left(x+a+U_t, y+b+V_t\right) \right|$$

Where $\mathrm{bound}(x, y)$ is the list of four tiles which surround pixel i, t_i is the tile index which we identify as producing the minimum residual error for pixel i, and I_0' and I_1' are the normalized grayscale images for tile-matching as defined in the main paper. With this we can produce a per-pixel flow/confidence field, where the per-pixel flow is simply the per-tile flow using these tile assignments, and the per-pixel confidence is the per-tile confidence attenuated by the per-pixel image residual.

$$\hat{u}_i = U_{t_i} \qquad \hat{v}_i = V_{t_i} \tag{5}$$

$$\hat{c}_i = C_{t_i} \exp\left(- \frac{\max\left(0, \left| I_0'(x,y) - I_1'(x+\hat{u}_i, y+\hat{v}_i) \right| - \epsilon_{up} \right)}{\sigma_{up}} \right)$$

Where $\sigma_{up} = 0.5$ and $\epsilon_{up} = 0.2$. Updating the per-pixel confidence in this way means that pixels which are not well explained by any nearby tile have very low confidence, and will therefore be inpainted during optimization.

1.5 Flow Asymmetry

Now that we have our per-pixel "forward" and "backward" flow fields for each image pair, we can reason about the symmetry or asymmetry these flow fields. If the estimated flow from pixel i in image 0 maps to pixel j in image 1, we would also expect the estimated flow from pixel j in image 1 to map back to pixel i in image 0. If this property does not hold, then the forward flow estimate at pixel i should not be trusted, and its confidence will be decreased accordingly. In a small abuse of notation, here let $\hat{u}_f(x, y) = u_i$ where pixel i is located at position (x, y) in our "forward" flow field, and let $\hat{u}_b(x, y) = u_i$ for our "backward" flow field (with $\hat{v}_f$, $\hat{v}_b$ defined similarly).

$$\begin{aligned} a_i &= \left(\hat{u}_f(x,y) + \hat{u}_b(x + \hat{u}_f(x,y), y + \hat{v}_f(x,y))\right)^2 \\ &\quad + \left(\hat{v}_f(x,y) + \hat{v}_b(x + \hat{u}_f(x,y), y + \hat{v}_f(x,y))\right)^2 \\ \hat{c}_i &\leftarrow \exp\left(- \frac{a_i}{\sigma_{sym}^2} \right) \hat{c}_i \end{aligned} \tag{6}$$

Where $\sigma_{sym} = 4$. The asymmetry measure a_i is squared Euclidean distance between the flow at pixel i and the the negative backward flow at the pixel in the alternate image that i maps to according to its estimated flow. This same update can also be applied to the "backward" flow.