**Introduction and motivation**

- Given a single image of a static 3D scene, this work solves two tasks: a **Exposing Image Forgery** and a **Depth-based Scene Segmentation** by recovering camera motion that occurred during exposure.
- Necessary, since it is not possible to detect using naked eye. Previous techniques could not handle 3D scenes containing motion blur.
- **High level idea**: Scene Depth, camera trajectory and motion-blur kernels are inter-related.
- **Challenge**: Knowledge of one is required to estimate the other.
- **Solution**: Discovered a consistency between horizontal and vertical projections of spatially-varying blur kernels within an image.

---

**Blurring Model**

<table>
<thead>
<tr>
<th>Pixel correspondences $\mathbf{x}$ and $\mathbf{x}_t$ at depth $Z$, after a transformation $[\Phi, \mathbf{T}]$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathbf{x}_t = \mathbf{K} \mathbf{R} \mathbf{k}^{-1} \mathbf{x} + \frac{\mathbf{K} \mathbf{T} \mathbf{x}}{Z}$</td>
</tr>
</tbody>
</table>

If $w_i$ denotes the fraction of time camera spent in position $i$, motion blurred image $B$ can be derived from focussed image $I$ as:

$$B[\mathbf{x}] = \sum_{i=1}^{N} w_i I[\mathbf{p}_{0_i}^{-1}(\mathbf{x})]$$ (2)

Similarly, PSF at $\mathbf{x}= (x, y)$ can be derived from a single point:

$$p(x, y) = \sum_{i=1}^{N} w_i \delta[\mathbf{p}_{0_i}^{-1}(\mathbf{x})]$$ (3)

In matrix form, it is equivalent to $p^{x}(x, y) = M^{x}(x, y)W_D$

---

**Consistency of $h_{length}$ and $v_{length}$**

- **Assumption**: Small angle of rotation $\phi$.
- If we pick any two points: $\mathbf{x}_i$ and $\mathbf{x}_j$, on a PSF, the difference in their spatial locations can be expressed in terms of the PSF’s pixel coordinate $x$ and $y$:

$$\Delta x_{ij} = \langle x^1 y^1 \rangle [ \frac{0}{(t_i^1 - t_j^1) f} ] \frac{1}{Z^1} + \frac{(\Delta \Phi)^{y}}{Z^1}$$ (4)

$$\Delta y_{ij} = \langle x^1 y^1 \rangle [ \frac{0}{(t_i^1 - t_j^1) f} ] \frac{1}{Z^1} + \frac{(\Delta \Phi)^{x}}{Z^1}$$ (5)

---

**Camera Trajectory Estimation**

- **Camera Trajectory estimation from local PSFs. Algorithm**: [3].
  $$p^{ext} = M^{ext}W_D$$ (6)
- **Minimize with sparsity constraint**.
  $$\| p^{ext} - M^{ext}W_b \|^2 + c \parallel W_b \parallel^1$$ (7)

---

**Matching PSFs at various depths**

Using depth and camera trajectory, we generate all possible PSFs at location $x$. If the pixel $x$ was actually situated at a different scene depth $D_i$, the PSF would be modified as follows:

$$p^{D_i}(x, a) = \sum_{x=1}^{N} w_i \delta[a - (\mathbf{p}_{0_i}^{-1}(\mathbf{x}) - x)]$$ (8)

Low cross-correlation between actual PSF and estimated PSFs $\rightarrow$ Region Spliced!

---

**Results**

- Finally, we utilize natural image texture segmentation [Mobahi, IJCV 2011] of the input image to obtain meaningful region boundaries.

---

**References**

