LiteFlowNet360 - Revisiting Optical Flow Estimation in 360 Videos

ICPR 2020 | PAPER ID : 2255
Texas State University
By
Keshav Bhandari, Ziliang Zong & Yan Yan
12/05/2020

MOTIVATION

Are optical flow representation for normal and 360 videos are same?

How to exploit existing architecture to compute optical flow for 360 videos?

Do we have labelled data?
Architecture - **LiteFlowNet360** is inspired by Lite-FlowNet. Feature Extractions Layers are replaced with “**Spherical Convolution Learnable Layers**” using transformer network. We did flow inference by projecting the feature map to **tangential plane, rather than learning inference/regularization layer.**
Approach

STAGE - 1
- TRANSFORM
- Representation Learning
- Kernel* Transformation

STAGE - 2
- INTERMEDIATE TRANSFORM
- Repr. Refinement
- 360 Data Augmentation
- Supervised Training
- Flow Correction

STAGE - 3
- FINAL REFINEMENT
- Domain Transfer
- 360 Video Data
- Warping Correction
- Self-Supervised Training
Representation Learning

Multiple variable size kernels are learned, these kernels are used to perform convolution in Spherical domain (that is in projected tangential planes).

\[
F_k(X) = \begin{cases} 
F_0(X) & : k = 0 \\
(F_k \circ F_{k-1})(X) & : n > k > 0 
\end{cases}
\]

(1)

\[
F'_k(X) = \begin{cases} 
F'_0(F_0(X), \Omega) & : k = 0 \\
(F'_k(F_k, \Omega) \circ F'_{k-1}(F_{k-1}, \Omega))(X) & : n > k > 0 
\end{cases}
\]

(2)

\[
Y_k = F_k(X), Y'_k = F'_k(F_k(X), \Omega)
\]

\[
L_k = \|Y'_k - Y_k\|^2
\]

This is done in tangential planes, so multiple Ls

\[
L'_k = \frac{1}{n_g} \sum_{i} L_k(\Omega(Y'_k, Y_k))
\]

(4)
Representation Refinement

INTERMEDIATE TRANSFORM

Repr. Refinement

360 Data Augmentation

Supervised Training

Flow Correction

360 Data Augmentation

Flow Corrections
Domain Transfer

**Fig. 5.** Final Refinement process. Network from second stage is extended to have two parallel weight sharing architecture.

\[
M_i = \begin{cases} 
0 & \text{if} |\Delta_i \rightarrow j| \leq \epsilon \\
1 & \text{otherwise}
\end{cases}
\]

\[
\tilde{O}_{i \rightarrow j} = M_i \odot ((1 - M_j) + \tilde{O}_{j \rightarrow i})
\]

\[
L_p = \sum_{i,j} \frac{\psi(I_i - I'_i) \odot (1 - O_{i \rightarrow j})}{\sum 1 - O_{i \rightarrow j}}
\]

*There is no boundary case for warping.*

*Warping in 2D domain doesn’t make sense*

*Warping must be done in spherical domain.*
Quantitative Results

<table>
<thead>
<tr>
<th>Model</th>
<th>Data</th>
<th>#Layers</th>
<th>$EPE$</th>
<th>$L_p^*$</th>
</tr>
</thead>
<tbody>
<tr>
<td>LiteFlowNet[9]</td>
<td>Sintel360</td>
<td>0</td>
<td>$\sim 6.35$</td>
<td>$\sim 1.30$</td>
</tr>
<tr>
<td>LiteFlowNet +[14]</td>
<td>Sintel360</td>
<td>$&gt; 4$</td>
<td>$\geq 17$</td>
<td>$\geq 3.06$</td>
</tr>
<tr>
<td>Ours, Stage-2</td>
<td>Sintel360</td>
<td>4</td>
<td>$\sim 6.35$</td>
<td>$\sim 0.70$</td>
</tr>
<tr>
<td>Ours, Final</td>
<td>Sintel360</td>
<td>4</td>
<td>$\sim 3.95$</td>
<td>$\sim 0.60$</td>
</tr>
</tbody>
</table>
Qualitative Results

Conclusion

Flow Using Cube-maps only

Cube map using Lite-Flownet Flow representations adapt to represent 360 properties

STAGE-2
(Equirectangular Projection)

OURS (Equirectangular Projection)
LiteFlowNet360 is more of a domain adaption approach to estimate optical flow for 360 Videos. The core idea is to light on fundamental considerations before exploiting off-the-shelf trained models.

<Conclusion>

Thank You!!