Notes: CUDA Implementation of Differentiable Image Warping

Zhiwei Xu

16 August 2021

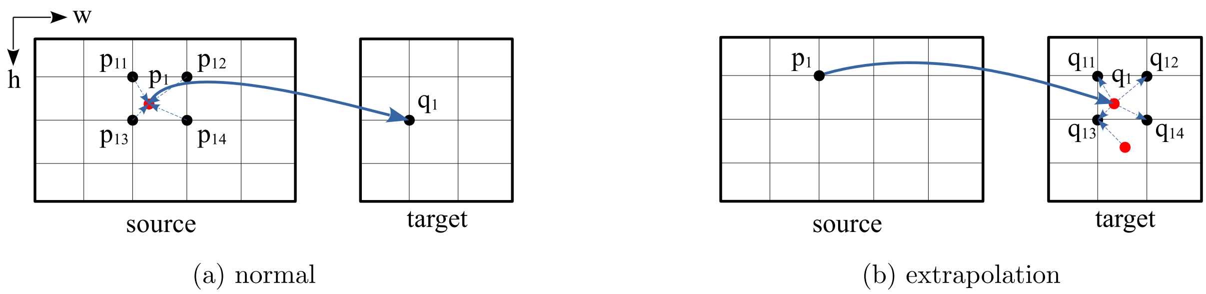

Figure 1: *Image warping. (a) can be achieved by torch.linspace() and torch.grid_sample() by Eq. (1). (b) is achieved by weighted sum of neighbours by Eq. (5), instead of rounding to the nearest neighbour, for the differentiability of coordinates between p_i and q_i.*

Given Fig. (1a), the forward and backward propagation are

$$f_t(q_i) = f_s(q_i + d_i) = f_s(p_i) = \sum_{j \in \mathcal{E}_i} g(p_i, p_{ij}) f_s(p_{ij}) \ , \tag{1a}$$

$$\begin{aligned} g(p_i, p_{ij}) &= g_h(p_i, p_{ij}) g_w(p_i, p_{ij}) \\ &= \left(1 - \frac{\left|h_{p_i} - h_{p_{ij}}\right|}{\left|h_{p_{ik}} - h_{p_{ij}}\right|}\right)\left(1 - \frac{\left|w_{p_i} - w_{p_{ij}}\right|}{\left|w_{p_{ik}} - w_{p_{ij}}\right|}\right) \ , \end{aligned} \tag{1b}$$

where $f_s(\cdot)$ and $f_t(\cdot) : \mathbb{R}^2 \mapsto \mathbb{R}$ are mapping functions for pixel intensity in source image and target image respectively, q_i and $d_i \in \mathbb{R}^2, k \in \mathcal{E}_i$, and $|k - j| = 2$. Generally, if image coordinates are in 1-step space, not sampled by such as torch.linspace(), $\left|h_{p_{ik}} - h_{p_{ij}}\right| = 1$ and $\left|w_{p_{ik}} - w_{p_{ij}}\right| = 1$. With loss L, the gradients of d_i follows the chain rule as follows,

$$\frac{dL}{dd_i} = \frac{\partial L}{\partial f_t(q_i)} \frac{\partial f_t(q_i)}{\partial d_i} = \frac{\partial L}{\partial f_t(q_i)} \frac{\partial f_t(q_i)}{\partial f_s(p_i)} \frac{\partial f_s(p_i)}{\partial d_i} = \sum_{j \in \mathcal{E}_i} f_s(p_{ij}) \frac{\partial L}{\partial f_t(q_i)} \frac{\partial f_t(q_i)}{\partial f_s(p_i)} \frac{\partial g(p_i, p_{ij})}{\partial d_i} \ , \tag{2}$$

where $\partial g(p_i, p_{ij}) / \partial d_i \in \mathbb{R}^2$ can be easily obtained.

Now, similarly, applying Eq. (1) to Fig. (1b),

$$f_t(q_i) = \sum_{j \in \mathcal{E}_i} g(q_i, q_{ij}) f_t(q_{ij}) \ , \tag{3}$$

and thus

$$\begin{aligned} f_t(q_{ij}) &= f_t(q_i) - \sum_{k \in \{\mathcal{E}_i \setminus j\}} g(q_i, q_{ik}) f_t(q_{ik}) \ , \\ f_t(q'_j) &= \frac{1}{|\mathcal{E}_j|} \sum_{i \in \mathcal{E}_j} f_t(q_{ij}) \ . \end{aligned} \tag{4}$$

With $\partial L/\partial f_t(q'_j)$, the calculation of dL/dd_i is difficult due to the nested $f_t(q_{ik})$ in $f_t(q_{ij})$.

Alternatively,

$$f_t(q'_j) = \frac{1}{\sum_{i \in \mathcal{E}_j} g(q_i, q_{ij})} \sum_{i \in \mathcal{E}_j} g(q_i, q_{ij}) f_t(q_i) \ , \tag{5}$$

$$\begin{aligned} \frac{dL}{dd_i} &= \sum_{j \in \mathcal{E}_i} \frac{\partial L}{\partial f_t(q'_j)} \frac{\partial f_t(q'_j)}{\partial d_i} = \sum_{j \in \mathcal{E}_i} \frac{\partial L}{\partial f_t(q'_j)} \frac{\partial f_t(q'_j)}{\partial g(q_i, q_{ij})} \frac{\partial g(q_i, q_{ij})}{\partial d_i} \\ &= \sum_{j \in \mathcal{E}_i} \frac{\partial L}{\partial f_t(q'_j)} \frac{f_t(q_i) - f_t(q'_j)}{\sum_{i \in \mathcal{E}_j} g(q_i, q_{ij})} \frac{\partial g(q_i, q_{ij})}{\partial d_i} \ . \end{aligned} \tag{6}$$

While one can decompose $f_t(q'_j)$ into $\sum_{i \in \mathcal{E}_j} g(q_i, q_{ij})$ and $\sum_{i \in \mathcal{E}_j} g(q_i, q_{ij}) f_t(q_i)$ to calculate partial derivatives on d_i separately, followed by an accumulation.

Note that the key of differentiating d_i which is a part of index in $f(\cdot)$ is to cast it as a weight using $g(\cdot, \cdot)$ instead of rounding it to the nearest neighbour.