快速引导滤波及其代码实现

Dezeming Family

2023 年 2 月 15 日

DezemingFamily 系列文章和电子书**全部都有免费公开的电子版**，可以很方便地进行修改和重新发布。如果您获得了 DezemingFamily 的系列电子书，可以从我们的网站 [https://dezeming.top/] 找到最新的版本。对文章的内容建议和出现的错误也欢迎在网站留言。

目录

一　论文基础介绍

我们再描述一下引导滤波要满足的约束关系。

我们假设在一个局部窗口 ω_k 内，像素颜色要满足局部线性模型 $q_i = a_k I_i + b_k$，每个窗 ω_k 内的 a_k 和 b_k 是一致的。定义每个 a_k 和 b_k 的损失函数：

$$E(a_k, b_k) = \sum_{i \in \omega_k} ((a_k I_i + b_k - p_i)^2 + \epsilon a_k^2) \tag{一.1}$$

一个像素会同时被多个窗口包含进去，每个窗口都能求出一个对应的 a_k 和 b_k，所以要把这些值取平均，作为该像素最终的 a_k 和 b_k。

这个约束可以用最小二乘法原理来计算（推导无需用矩形形式），对 E 求偏导：

$$\begin{aligned} \frac{\partial E}{\partial a_k} &= 2 \sum_{i \in \omega_k} ((a_k I_i + b_k - p_i) I_i + \epsilon a_k) \\ \frac{\partial E}{\partial b_k} &= 2 \sum_{i \in \omega_k} ((a_k I_i + b_k - p_i)) \end{aligned} \tag{一.2}$$

让偏导数为 0，先表示出 b_k，然后代入到 a_k 中，就能得到对应的表示形式了。最终输出的 a 和 b 是要取均值的：

$$q_i = \overline{a}_i I_i + \overline{b}_i \tag{一.3}$$

我们不能让窗口过大，否则就不满足局部线性模型了。

二　快速算法

我们通过下采样的方法来提高效率。我们看到最后的结果，$\overline{a}_i$ 和 $\overline{b}_i$ 是取了均值，因此没有必要在全分辨率上执行。

我们用双线性方法下采样原图 p 和引导图 I，下采样率为 s，然后计算得到的 $\overline{a}$ 和 $\overline{b}$，之后再用双线性方法上采样到原始尺寸。

Algorithm 2 Fast Guided Filter.

1: $I' = f_{\text{subsample}}(I, s)$
$p' = f_{\text{subsample}}(p, s)$
$r' = r/s$
2: $\text{mean}_I = f_{\text{mean}}(I', r')$
$\text{mean}_p = f_{\text{mean}}(p', r')$
$\text{corr}_I = f_{\text{mean}}(I'. * I', r')$
$\text{corr}_{Ip} = f_{\text{mean}}(I'. * p', r')$
3: $\text{var}_I = \text{corr}_I - \text{mean}_I . * \text{mean}_I$
$\text{cov}_{Ip} = \text{corr}_{Ip} - \text{mean}_I . * \text{mean}_p$
4: $a = \text{cov}_{Ip} ./ (\text{var}_I + \epsilon)$
$b = \text{mean}_p - a. * \text{mean}_I$
5: $\text{mean}_a = f_{\text{mean}}(a, r')$
$\text{mean}_b = f_{\text{mean}}(b, r')$
6: $\text{mean}_a = f_{\text{upsample}}(\text{mean}_a, s)$
$\text{mean}_b = f_{\text{upsample}}(\text{mean}_b, s)$
7: $q = \text{mean}_a . * I + \text{mean}_b$

我们发现当 $s = 4$ 的时候，跟原图相比也没有很明显的区别。

三　代码实现

代码实现过程也比较简单：

```python
import cv2
import numpy as np
def guided_filter(I,p,s,win_size,eps):
    # 下采样，默认使用双线性插值法
    I_ = cv2.resize(I, (int(I.shape[1]/s), int(I.shape[0]/s)))
    p_ = cv2.resize(p, (int(p.shape[1]/s), int(p.shape[0]/s)))
    # 加1为了防止窗小于1
    win_size = int(win_size / s + 1)

    mean_I = cv2.blur(I_,(win_size,win_size))
    mean_p = cv2.blur(p_,(win_size,win_size))

    corr_I = cv2.blur(I_*I_,(win_size,win_size))
    corr_Ip = cv2.blur(I_*p_,(win_size,win_size))

    var_I = corr_I-mean_I*mean_I
    cov_Ip = corr_Ip - mean_I*mean_p

    a = cov_Ip/(var_I+eps)
    b = mean_p-a*mean_I

    mean_a = cv2.blur(a,(win_size,win_size))
    mean_b = cv2.blur(b,(win_size,win_size))

    # 上采样，默认用双线性插值法
    mean_a = cv2.resize(mean_a, (I.shape[1], I.shape[0]))
    mean_b = cv2.resize(mean_b, (I.shape[1], I.shape[0]))

    q = mean_a*I + mean_b
    return q
```

使用方法如下：

```python
image = cv2.imread('noisyImage.png', cv2.IMREAD_COLOR) / 255.0
q = guided_filter(image, image, 4, 8, 0.14) * 255.0
cv2.imwrite('output.png', q)
```

参考文献

[1] K. He, J. Sun, and X. Tang. Guided image filtering. In ECCV, pages 1–14. 2010.

[2] He, Kaiming, Jian Sun, and Xiaoou Tang. "Guided image filtering." IEEE transactions on pattern analysis and machine intelligence 35.6 (2012): 1397-1409.

[3] He, Kaiming, and Jian Sun. "Fast guided filter." arXiv preprint arXiv:1505.00996 (2015).