Joint Super-Resolution and Optical Flow Estimation

1 Energy Functional

Let $\Omega_l \subset \mathbb{R}$ be a discretized rectangular domain with $w \times h$ pixels and $\Omega_h \subset \mathbb{R}$ a domain with $W \times H$ pixels. Let $f_1, \cdots, f_n : \Omega_l \to \mathbb{R}^k$ be low resolution color input images with k color channels. We seek to jointly estimate high resolution images $u_1, \cdots, u_n : \Omega_h \to \mathbb{R}^k$ and optical flow fields $v_1, \cdots, v_{n-1} : \Omega_h \to \mathbb{R}^2$ from the low resolution images.

We model the problem in terms of energy minimization of the following functional:

$$\begin{aligned} E(u,v) = \sum_{i=1}^{n} \sum_{x \in \Omega_l} \alpha \|Au_i - f_i\|_1 + \beta TV(u_i) + \gamma \sum_{i=1}^{n-1} \sum_{x \in \Omega_h} ||u_i(x) - u_{i+1}(x + v_i(x))||_1 \\ + TV(v_i^1) + TV(v_i^2) \end{aligned} \tag{1}$$

Here $A : (\Omega_h \to \mathbb{R}^k) \to (\Omega_l \to \mathbb{R}^k)$ is a linear operator which maps a high-resolution image to a low resolution image by blurring it with a gaussian kernel and downsampling it.

$TV(u) := \sum_{x \in \Omega} \|\nabla u(x)\|_2$ denotes the TV regularizer.

2 Optimization

Since the energy is hard to minimize jointly in u and v we employ a block-coordinate descent approach:

$$\begin{aligned} v^{k+1} &= \arg\min_v E(u^k, v^k), \\ u^{k+1} &= \arg\min_u E(u^k, v^{k+1}). \end{aligned} \tag{2}$$

2.1 Solving the Problem in v (TV-L1 Optical Flow).

For fixed u^k the problem reads:

$$v^{k+1} = \arg\min_v \sum_{i=1}^{n-1} \gamma \sum_{x \in \Omega_h} ||u_i^k(x) - u_{i+1}^k(x + v_i(x))||_1 + TV(v_i^1) + TV(v_i^2) \tag{3}$$

For simplicity, we first consider the case $n = 2$:

$$v^{k+1} = \arg\min_{v} \gamma \sum_{x \in \Omega_h} ||u_1^k(x) - u_2^k(x + v(x))||_1 + TV(v^1) + TV(v^2) \tag{4}$$

This energy is nonconvex in v, due to the first term. Thus we linearize it using the first order Taylor expansion,

$$||u_1^k(x) - u_2^k(x + v(x))||_1 \approx ||u_1^k(x) - u_2^k(x) - \nabla u_2^k(x)^T v(x)||_1,$$

and end up at the following convex problem:

$$v^{k+1} = \arg\min_{v} \gamma \sum_{x \in \Omega_h} || \underbrace{u_1^k(x) - u_2^k(x)}_{=:-b(x)} - \underbrace{\nabla u_2^k(x)^T v(x)}_{=:(Av)(x)} ||_1 + TV(v^1) + TV(v^2), \tag{5}$$

where $v = (v^1, v^2)$.

2.1.1 Primal-Dual Optimization

Since the energy is non-differentiable, a gradient descent based approach does not work. We employ the primal-dual algorithm described in [**?**, **?**] to minimize the energy. First, we rewrite (**??**) as an equivalent saddle-point problem:

$$\min_{v} \max_{p \in C, q_1 \in D, q_2 \in D} \langle p, Av + b \rangle + \langle q_1, \nabla v^1 \rangle + \langle q_2, \nabla v^2 \rangle \tag{6}$$

The update equations for the algorithm then read:

$$\begin{aligned}
p^{k+1}(x) &= \operatorname*{proj}_{C}(p^k(x) + \sigma_p(x)((Av^k)(x) + b(x))) \\
q_1^{k+1}(x) &= \operatorname*{proj}_{D}(q_1^k(x) + \sigma_q(\nabla v_1^k)(x)), \\
q_2^{k+1}(x) &= \operatorname*{proj}_{D}(q_2^k(x) + \sigma_q(\nabla v_2^k)(x)), \\
\bar{p}^{k+1}(x) &= 2p^{k+1} - p^k, \\
\bar{q}_1^{k+1}(x) &= 2q_1^{k+1} - q_1^k, \\
\bar{q}_2^{k+1}(x) &= 2q_2^{k+1} - q_2^k, \\
v^{k+1}(x) &= v^k - \tau(x) \left((A^T \bar{p}^{k+1})(x) - \begin{pmatrix} (\operatorname{div} \bar{q}_1^{k+1})(x) \\ (\operatorname{div} \bar{q}_2^{k+1})(x) \end{pmatrix} \right).
\end{aligned} \tag{7}$$

The sets C and D are defined as

$$\begin{aligned}
C &= \{x \in \mathbb{R} \mid |x| \leq \gamma\}, \\
D &= \{x \in \mathbb{R}^{2n_c} \mid \|x\|_2 \leq 1\},
\end{aligned} \tag{8}$$

and the projections proj_C, proj_D can be implemented as an orthogonal projection on a sphere. Only project if you lie outside of the constraint.

The step sizes are chosen according to the scheme described in [**?**] (see Lemma 2, equation 10, we set $\alpha = 1$):

$$\begin{aligned} \sigma_p(x) &= \frac{1}{\sum_j |A(x,j)|}, \\ \sigma_q &= \frac{1}{2}, \\ \tau(x) &= \frac{1}{2 + 2 + \sum_i |A(i,x)|}, \end{aligned} \tag{9}$$

where $A(x, j)$ denotes the element in row x and column j.

Allocate memory for the variables $p \in \mathbb{R}^{w*h*n_c}, q_1 \in \mathbb{R}^{w*h*2*n_c}, q_2 \in \mathbb{R}^{w*h*2*n_c}, \bar{p}, \bar{q}_1, \bar{q}_2, v \in \mathbb{R}^{w*h*2*n_c}$ as `float` arrays and implement CUDA kernels for the update equations of the primal-dual algorithm. One kernel should perform the update in p, q_1, q_2 and do the overrelaxation, the other kernel should do the update in v.

2.2 Solving the Problem in u.

For fixed v^{k+1} the problem is simplified to:

$$u^{k+1} = \arg\min_u \sum_{i=1}^{n} \alpha ||Au_i - f_i||_1 + \beta TV(u_i) + \gamma \sum_{i=1}^{n-1} \sum_{x \in \Omega_h} ||u_i(x) - u_{i+1}(x + v_i^{k+1}(x))||_1 \tag{10}$$

Using the first order Taylor expansion for linearization (like in the optical flow optimization) and considering the more specific case $n = 2$ we get:

$$\begin{aligned} u^{k+1} = \arg\min_u \; & \alpha ||Au_1 - f_1||_1 + \alpha ||Au_2 - f_2||_1 + \beta TV(u_1) + \beta TV(u_2) \\ & + \gamma \sum_{x \in \Omega_h} ||u_1(x) - u_2(x) - \nabla u_2(x)^T v^{k+1}(x))||_1 \end{aligned} \tag{11}$$

2.2.1 Primal-Dual Optimization

Again the energy is non-differentiable and therefore we use the above mentioned primal-dual algorithm here as well. The corresponding saddle-point problem can be formulated as:

$$\min_{\substack{u_1,u_2 \\ q_1 \in E \\ q_2 \in E \\ r \in G}} \max_{\substack{p_1 \in F \\ p_2 \in F}} \sum_{i=1}^{2} \langle p_i, Au_i - f_i \rangle + \langle q_i, \nabla u_i \rangle \; + \; \langle r, B_{flow} u \rangle \tag{12}$$

where $u = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}$ and $B_{flow} = \begin{pmatrix} I, & -I - v^1 \partial_x - v^2 \partial_y \end{pmatrix}$

The resulting update equations are:

$$\begin{aligned}
p_i^{k+1} &= \operatorname*{proj}_F(p_i^k + \sigma_p(Au_i^k - f_i)),\\
q_i^{k+1} &= \operatorname*{proj}_E(q_i^k + \sigma_q(\nabla u_i)),\\
r^{k+1} &= \operatorname*{proj}_G\left(r^k + \sigma_r B_{flow}\begin{pmatrix}u_1^k\\ u_2^k\end{pmatrix}\right),\\
\bar{q}_i^{k+1} &= 2q_i^{k+1} - q_i^k,\\
\bar{p}_i^{k+1} &= 2p_i^{k+1} - p_i^k,\\
\bar{r}^{k+1} &= 2r^{k+1} - r^k\\
u_i^{k+1} &= u_i^k - \tau_i(A^T\bar{p}_i^{k+1} - div(\bar{q}_i^{k+1}) + \underbrace{(B_{flow}^T\bar{r}^{k+1})_i}_{s_i}),
\end{aligned} \tag{13}$$

For implementation issues the operator B_{flow} can be further decomposed:

$$r^{k+1}(x) = \operatorname*{proj}_G(r^k(x) + \sigma_r(u_1^k(x) - u_2^k(x) - (\partial_x u_2^k v^1)(x) - (\partial_y u_2^k v^2)(x)) \tag{14}$$

In the same way we can apply B_{flow}^T having dirichlet boundary conditions for the derivatives:

$$s_i = \begin{cases}\bar{r}^{k+1} & i = 1\\ -\bar{r}^{k+1} + v^1\partial_x^-\bar{r}^{k+1} + v^2\partial_y^-\bar{r}^{k+1} & i = 2\end{cases} \tag{15}$$

The used sets for the dual variables are defined as:

$$\begin{aligned}
E &= \{x \in \mathbb{R}^{2n_c} \mid \|x\|_2 \le \beta\},\\
F &= \{x \in \mathbb{R} \mid |x| \le \alpha\},\\
G &= \{x \in \mathbb{R} \mid |x| \le \gamma\}
\end{aligned} \tag{16}$$

Finally the step sizes for the update steps are chosen like this:

$$\begin{aligned}
\sigma_q &= \frac{1}{2},\\
\sigma_p &= 1,\\
\sigma_r(x_{(ijc)}) &= \frac{1}{2 + 2|v^1(x_{(ij)})| + 2|v^2(x)|}\\
\tau_i &= \frac{1}{1 + 4 + t_i(x)},\\
t_i(x) &= \begin{cases}1 & i = 1\\ 1 + |v^1(x)| * 2 + |v^2(x)| * 2 & i \ge 2\end{cases}
\end{aligned} \tag{17}$$

For completeness and for debugging reasons it follows a short more detailed analysis of the used operators:

$$
\begin{aligned}
B_{flow} &= \left(I, \quad -I - v^1\partial_x - v^2\partial_y\right), \\
B_{flow}^T &= \begin{pmatrix} I \\ -I + \underset{dirichlet^x}{\partial_x^-} v^1 + \underset{dirichlet^y}{\partial_y^-} v^2 \end{pmatrix}, \\
A &= DB_l, \\
A^T &= B_l D^T
\end{aligned}
\tag{18}
$$

$$
\begin{aligned}
A &\colon \mathbb{R}^{W*H*c} \mapsto \mathbb{R}^{w*h*n_c}, \\
B_{flow} &\colon \mathbb{R}^{2*W*H*c} \mapsto \mathbb{R}^{W*H*n_c}, \\
B_l &\colon \mathbb{R}^{W*H*c} \mapsto \mathbb{R}^{W*H*n_c}, \\
D &\colon \mathbb{R}^{W*H*c} \mapsto \mathbb{R}^{w*h*n_c}
\end{aligned}
$$