BP Denoising

raluca.scona

June 2020

1 Robust Measurement Factor

The state of pixel x is y: $x = (y)$, where x is the pixel ID and y is the pixel value. The precision Λ and measurement z are scalars. The measurement function is:

$$h(x) = y \tag{1}$$

And the Jacobian:

$$J = \frac{\partial h(x)}{\partial x} = \frac{\partial y}{\partial y} = 1 \tag{2}$$

When linearising:

$$\eta = J^T\Lambda(Jx_0 + z - h(x_0)) = J^T\Lambda(y_0 + z - y_0) = J^T\Lambda z \tag{3}$$

$$\Lambda' = J^T\Lambda J \tag{4}$$

Notice that neither of these terms depend on the state, so they always have the same linearisation form.

When choosing the N_σ threshold for Huber, we can first analyse the Mahalanobis distance:

$$M_s = \sqrt{(z-h(x))^T\Lambda(z-h(x))} = |z-y|\sqrt{\Lambda} = |z-y|\frac{1}{\sigma} \tag{5}$$

So, for example, I can choose $N_\sigma = 1\sqrt{\Lambda}$. In my opinion it is a good idea to define this as a function of Λ because the scalar next to it will then represent a distance in pixel space.

When uncertainty is high, σ is large, meaning that $\frac{1}{\sigma}$ is small. In this setup, this results in having a stricter threshold N_σ for higher uncertainties. However we should probably also have an upper bound for N_σ in case uncertainty is too low, so perhaps something like $N_\sigma = \min(\alpha\sqrt{\Lambda}, \epsilon)$ could work well in practice.

My main concern with not having a dependence on $\sqrt{\Lambda}$ is that if uncertainty is set very high, large errors may be below the threshold and not treated using the L1 term.

2 Robust Smoothness Factor

This is a function of two pixels, $(x_1, x_2)^T$, whose states are $(y_1, y_2)^T$. The precision Λ is a scalar and the measurement $z = 0$. The measurement function is:

$$h(x_1, x_2) = y_2 - y_1 \tag{6}$$

And the Jacobian:

$$J = \left(\frac{\partial h(x_1, x_2)}{\partial x_1}, \frac{\partial h(x_1, x_2)}{\partial x_2}\right) = \left(\frac{\partial(y_2 - y_1)}{\partial y_1}, \frac{\partial(y_2 - y_1)}{\partial y_2}\right) = (-1, 1) \tag{7}$$

When linearising:

$$\eta = J^T\Lambda(J(x_1, x_2) + z - h(x_1, x_2)) = J^T\Lambda((-1, 1)(y_1, y_2)^T + 0 - (y_2 - y_1)) = J^T\Lambda 0 = (0, 0)^T \tag{8}$$

$$\Lambda' = J^T\Lambda J \tag{9}$$

Where Λ' is a 2 x 2 matrix. These quantities also do not depend on the current state values so this linearisation always has the same form. The Mahalanobis distance:

$$M_s = \sqrt{(z - h(x_1, x_2))^T\Lambda(z - h(x_1, x_2))} = |y_1 - y_2|\sqrt{\Lambda} \tag{10}$$

For this I also choose $N_\sigma = 1\sqrt{\Lambda}$ as explained above.