Manual for Depth Map Automatic Generator 9b (DMAG9b)

Ugo Capeto*

December 4, 2016

DMAG9 is an implementation of Fast Bilateral Solver [2] which is itself a derivation of Fast Bilateral-Space Stereo [1].

DMAG9b can be used to improve the quality of (noisy) depth maps. DMAG9b maintains the depths of pixels that have high confidence and smoothes (in an edge-preserving way) the depths of pixels that have low confidence. The confidence map is computed automatically from the input depth map by linking the depth variance at a pixel to the confidence (placed on the depth at that pixel): A high variance indicates a low confidence while a low variance indicates a high confidence. DMAG9b uses the Domain Transform [3] to compute the variance.

Assuming you have a reference image and an input depth map in a directory somewhere on your computer, the files `dmag9b.bat` and `dmag9b_input.txt` must be copied to that directory. The file `dmag9b.bat` must be modified so that the path to `dmag9b.exe` is the correct one for your computer. To run DMAG9b, you simply double-click on `dmag9b.bat`.

The format of `dmag9b_input.txt` is as follows:

```
Reference image
Input depth map
Spatial sample rate
Range (color) sample rate
Lambda
Hash table size
Number of PCG (Preconditioned Conjugate Gradient) iterations
Scale parameter of Geman-McClure function
Number of IRLS (Iteratively Reweighted Least Squares) iterations
Radius (confidence map)
Gamma proximity gamma_p (confidence map)
Gamma color similarity gamma_c (confidence map)
Sigma (confidence map)
Output depth map
```

The image names cannot have spaces in them. They should not be in full path format (like `C:\this_dir\that_dir\image.png`). I also wouldn't use a point in the body of the

*e-mail: `ugocapetodifrancia3d@gmail.com`

name (`image_left.png` is ok but `image.left.png` is not). For the reference image and the output depth map, the supported image file formats are tiff, png, and jpeg. For the input depth map, the supported file formats are tiff and png.

The reference image is (square) gridded and the dimension of each square cell is equal to the spatial sample rate. For example, if the spatial sample rate is 8, each grid cell is 8x8 pixel square. All the pixels contained in a given cell will have the same depth/disparity (the one given to the cell by the program). If the spatial sample rate is set to an extremely large value, like the image size, the depth/disparity map is going to be one solid color as all pixels will have the same depth/disparity. If set to a large value, say 256, the disparity map will look very blocky, each block being 256x256 pixels. As you go down in value (256, 128, 64, 32, 16, 8, 4, 2, 1), the blocks become smaller and smaller until the blocks are actually pixels (spatial sample rate equal to 1). Good choices are 4, 8, 16, and 32.

Each color channel (there are three of them) is gridded and the size of each cell in that dimension is equal to the range (color) sample rate. If set to its maximum value (256), the depths/disparities appear smoothed/blurred without considering color variations (not edge preserving at all). In other words, it behaves like a classic blur and the object boundaries appear quite fuzzy. As you go down in value (256, 128, 64, 32, 16, 8, 4, 2, 1), the smoothing of the depths/disparities appears to be more and more color aware, that is, the blur occurs within areas which are more and more similar in color. Good choices are 4, 8, 16, and 32.

Lambda controls the smoothness of the output depth map. The lower lambda is, the less smooth (in an edge preserving sense) the output depth map is going to be.

The hash table size dictates the speed at which the bilateral space vertices are created. The higher the number, the faster it is going to be at the expense of memory usage. A good value is 10000 but it should be increased (say, to 100000) if the number of bilateral space vertices per hash table bucket becomes too large (say, more than 100).

The number of PCG iterations controls the number of iterations performed by Fast Bilateral Solver when solving the linear system of equations. The larger the number, the more accurate but at the expense of speed.

The Geman-McClure scale parameter is used in the evaluation of the confidence values for each pixel during each iteration of IRLS. It's a parameter that makes sense only if the IRLS number of iterations is greater than 1 (Fast Bilateral Solver switches to Robust Bilateral Solver).

If the number of IRLS iterations is greater than 1, Fast Bilateral Solver is iterated upon to make it less sensitive to outliers in the input depth map (The process is then called Robust Bilateral Solver.) The IRLS number of iterations controls how many times Fast Bilateral Solver is iterated upon by Robust Bilateral Solver. As usual, the higher the number, the more accurate but at the expense of time.

The confidence map `radius` is the radius of the window used to compute the depth expectation at a given pixel. That depth expectation is used to compute the depth variance at that pixel which is in turn used to compute the confidence in the depth at that pixel.

The confidence map `gamma_p` is the bandwidth of the bilateral filter weight that relates to space. The smaller `gamma_p` is, the less weight is given to pixels in the window (defined by `radius`) that are far away in pixel space from the pixel of reference.

The confidence map `gamma_c` is the bandwidth of the bilateral filter weight that relates to the range (color). The smaller `gamma_c` is, the less weight is given to pixels in the window (defined by `radius`) that are far away in color space from the pixel of reference.

The confidence map `sigma` is used to convert the depth variance to a confidence value (near 0 is low confidence and near 1 is high confidence). The lower `sigma` is, the less confidence will be given to pixels.

Because the Robust Bilateral Solver is used when there are possibilities of outliers in the input depth map (the case here), I would keep the number of IRLS iterations to 32 and the number of PCG iterations to something like 25. Of course, those values can be changed to whatever works best.

For the confidence map, for images in the 2 to 4 megapixels range, I would recommend to use 12 for `radius`, 12.0 for `gamma_p`, 12.0 for `gamma_c`, and a value ranging from 0.25 to 2.0 for `sigma`. The confidence map is saved as `con_map.tiff` by the program when it's running so you can have an idea of the depth confidences given to pixels. Black areas indicate low confidence while white areas indicate high confidence.

If you cannot scroll to the beginning of the command window (DMAG9b prints out useful info as it runs), you need to increase the screen buffer size of the command window by 1) right-clicking on the icon in the upper left and then clicking on Properties, 2) clicking on the Layout tab, 3) increasing the Screen Buffer Size Height to the maximum (9999), and 4) clicking OK. You should now be able to get to the beginning of the printouts next time you run DMAG9b.

References

[1] Jonathan T Barron, Andrew Adams, YiChang Shih, and Carlos Hernández. Fast bilateral-space stereo for synthetic defocus. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4466–4474, 2015.

[2] Jonathan T. Barron and Ben Poole. The fast bilateral solver. *CoRR*, abs/1511.03296, 2015.

[3] Eduardo SL Gastal and Manuel M Oliveira. Domain transform for edge-aware image and video processing. In *ACM Transactions on Graphics (TOG)*, volume 30, page 69. ACM, 2011.