Computer Vision Challenge

G22:

Fengyi Wang 03710714, Shuo Ma 03717403, Wenjian Zhao 03709772
Cong Wang 03718194, Chengjie Yuan 03714956

July 18, 2019

Contents

1 Algorithm

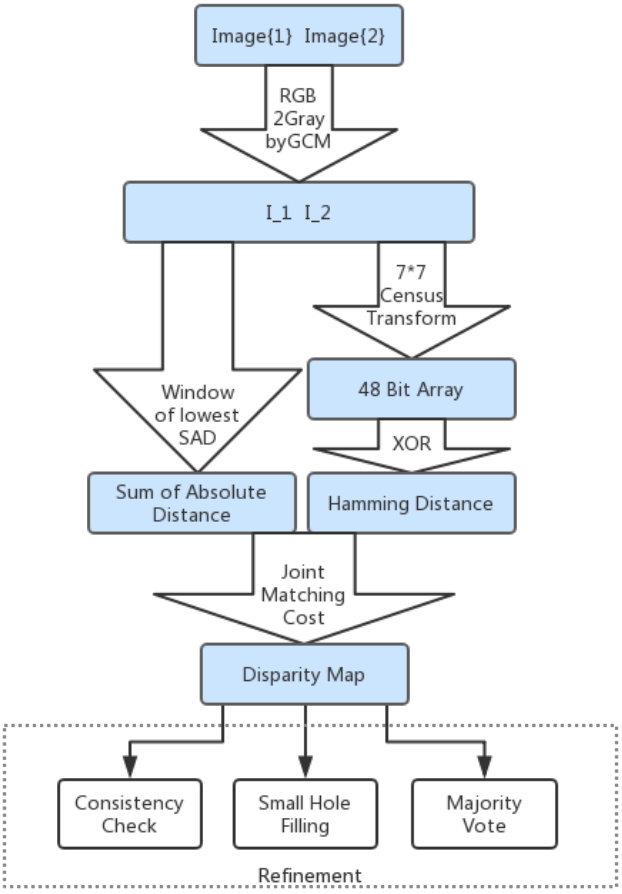


Figure 1: total algorithm flow chart

We have two methods to realize our algorithm by using two kinds of windows. One is with fixed window size and the other with a variable window size. In practice we use this two methods and compare the results of them. At last we choose one that works better in certain test.

1.1 Census transform

Census Transform method is used as matching cost function. It's robust, when the difference of absolute intensity value are compared. In addition, It is hardly affected, when the luminance value of each angle are different.
The main idea of Census transform is to build a p-centered window for a random pixel p, and generate:

$$C_{census}(I(p)) = \otimes_{p' \in N(p)} \quad \xi(p, p') \tag{1}$$

$$\xi(p, p') = \begin{cases} 1 & I(p) < I(p') \\ 0 & otherwise \end{cases} \tag{2}$$

Where the symbol $\otimes$ represents concatenation, N(p) denotes the non-parametric window around P.
The following example shows the census transform. It consists of the digits 0 and 1. The value of center pixel is a standard. When the other value less than the standard, set to 0. In contrast it will be 1. With help of Hamming distance, it can be compared, whether the pixels of two image are similar. Then the matrix are turned into bit array. Through an exclusive OR it can be figured out, how similar two pixel are[3].

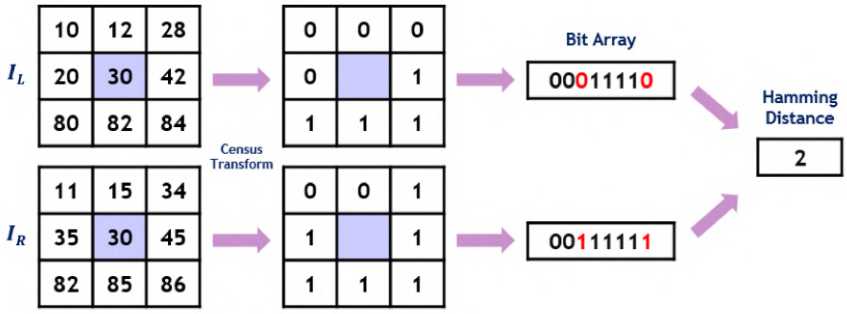


Figure 2: Census transform from moving window of stereo image

1.2 Integral image

Integral image is an efficient method which we use to calculate some rectangular areas of image. At each location x,y of the integral image consists of sum of the pixels above and left to x,y in the orginal image[4], namely:

$$image\_integral(x,y) = \sum_{x' \leq x, y' \leq y} image(x', y') \tag{3}$$

The integral image can be computed in linear time for all x, y in the original image. Using the following recurrences:

$$s(x,y) = s(x,y-1) + image(x,y) \tag{4}$$

$$image\_integral(x,y) = image\_integral(x-1,y) + s(x,y) \tag{5}$$

where image(x,y)is the original image, image_integral(x,y) is the integral image, s(x,y) is the cumulative sum row sum, s(x,-1)=0,image_integral(-1,y)=0.
After generating the integral image, any rectangular sum can be computed in four array references. As can be seen from an example in Figure 1 [5].

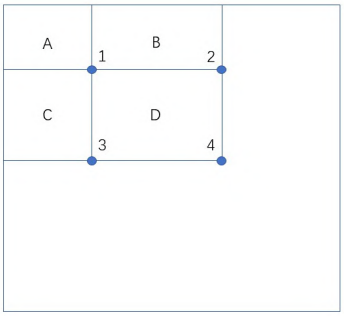


Figure 3: rectangular area computation

The number 1 contains the value of the sum of pixels in rectangular area A, At the location 2 the value

is A+B, similarly the values at 3 and 4 are A+B+C and A+B+C+D respectively. Hence the sum of pixels within the rectangular area D can be computed with these four array references. The value of D is 4+1-2-3.

1.3 Adaptive window

Since fixed window algorithms clearly do not perform well, there has been some work on varying window size and/or shape. Such variable window methods face two main issues. First is designing an appropriate window cost, since windows of different sizes and/or shapes are to be compared. Second is efficiently searching the space of possible windows for the one with the best cost[5].
We can define our window cost $C_d(W)$ as following:

$$C_d(W) = \overline{e} + \frac{\beta}{\sqrt{W} + \gamma} \tag{6}$$

Here W is a rectangular set of pixels, and d is some disparity, since a window is evaluated at some disparity. $\overline{e}$ is just the average measurement error in the window:$\overline{e} = \frac{\sum_{(x,y)\in W} e_d(x,y)}{|W|}$,The inclusion of this term is obvious: the lower the measurement error, the more likely disparity d is for pixels in W. We normalize by window size $|W|$ since we will be comparing windows of different sizes. β, γ are parameters assigning relative weights to terms in equation(6).

1.4 Joint match cost

Matching cost is the metric function to judge whether two pixels are similar, and it's the only foundation for estimation of disparity.Therefore a good matching cost function plays a very important role in improving the algorithm's robustness in different scenes. Moreover,considering that the commonly used AD(absolute distance) cost function has better matching effects in areas with smooth textures. A joint match cost adopts the matching cost function that combines AD and Census transform(equation (1)) together.For a random pixel p in the left view, establish a p-centered 7x7 window and define the matching cost of p as follows[6]:

$$J(p,d) = \rho(D_{census}(p,d), \lambda_{census}) + \rho(D_{AD}(p,d), \lambda_{AD}) \tag{7}$$

Here d is the presupposed disparity value, D_{census} and D_{AD} are the matching costs based on Census transform and AD respectively, $\rho(D,\lambda)$ is a normalized function, with the following computing formula:

$$D_{census}(p,d) = D_{Hamming}(C_{census}(I_L(x_p,y_p)), C_{census}(I_R(x_p-d,y_p))) \tag{8}$$

$$D_{AD}(p,d) = \min\{|I_L(x_p,y_p) - I_R(x_p-d,y_p)|, \tau\} \tag{9}$$

$I_L(x,y)$ and $I_R(x,y)$ refer to the gray values of corresponding pixels in left and right view respectively, C_{census} is calculated from Formula (1), λ_{census}, λ_{AD} and τ are prior parameters. In our project λ_{census} and λ_{AD} are set as a fixed value, i.e half of D_{census} and D_{AD} respectively.

1.5 Refinement

1.5.1 Consistency check

Occlusions create points that do not belong to any conjugate pairs. In many cases, occlusions occur at depth discontinuities: indeed, one may observe hat occlusions on one image correspond to disparity jumps on the other. Although evidences have been reported that occlusions help the human visual system in detecting object boundaries, in computational stereo they are a major source of errors.
A key observation to address the occlusion problem is that matching is not a symmetric process: when searching for conjugate pairs, only the visible points in one image are matched. If the role of left and right images is reversed, new conjugate pairs are found. The so-called left-right consistency constraint states that feasible conjugate pairs are those found with both direct and reverse matchings. It is worthwhile noting that the latter is equivalent to the uniqueness constraint, which states that each point on one image can match at most one point on the other image. Consider for instance an occluded point, e.g., B, in the

left image of Fig. 2: although it has no corresponding point in the right image, the SSD minimisation matches it to some point (C‘) anyhow. One can see that the latter point, in turn, corresponds to a different point in the left image, but this information is available only by searching from right to left[1].
In our approach, occlusions are detected by checking the left-right consistency, and suppressing unfeasible

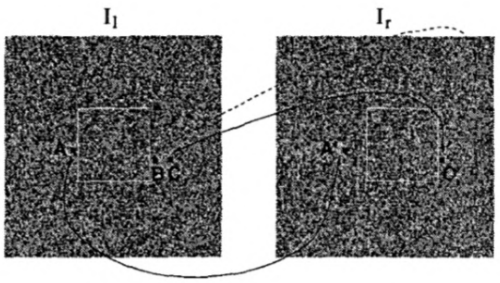


Figure 4: Left-right consistency. Point B which is occluded, is given C' as a match, but C' matches C != B. The pair (B,C') can be suppressed

matches accordingly. For each point on the left image the disparity dl(x) is computed as described in The SSD Correlation Algorithm. The process is repeated after reversing the two images. If $d_l(x)= -d_r(x + d_l(x))$ the point keeps its computed left disparity, otherwise it is marked as occluded and a disparity is assigned heuristically: following , we assume that occluded areas, occurring between two planes at different depth, take the disparity of the deeper plane.

1.5.2 Majority vote

This paper[7] presents a leading local algorithm and then accelerate it by parallel computing. The matching accuracy is increased because of cost aggregation over shape-adaptive support regions and disparity refinement using reliable initial estimates. The refinement and the restoration are jointly realized by a local voting method.
After cost aggregation, initial disparity estimates are selected using a WTA method. The errors of initial estimates mainly lie in homogeneous regions and occluded regions. In homogeneous regions, the aggregated matching costs at different disparities are usually similar and lack of discriminative power, therefore yielding noisy results. In the occluded regions, since there is no correspondence for the pixels in the other view, a local WTA framework is difficult to handle these pixels. In our assignment, we use this voting method to make the algorithm robust and more accurate.
The reliability of initial estimates is determined by a left-right consistency check, i.e., the disparity of p in the left image should have the similar absolute value as the disparity of the point $(x_p - d_p, y_p)$ in the right image. If the disparity estimate d_p passes the consistency check, it is labeled as reliable $\eta(d_p = 1)$, otherwise it is labeled as unreliable $\eta(d_p = 0)$.
We refine all the initial estimates with a similar local voting method. At each pixel p, we build a histogram $\varphi_p(.)$ over the reliable disparities in its support region as follows:

$$\varphi_p(d) = \sum_{s \in U(p)} \delta(d_s - d)\eta(d_s) \tag{10}$$

where $\delta(.)$ is an impulse function. The disparity d_p^* associated with the peak of the histogram is taken as the optimal disparity for p as follows:

$$d_p^* = \operatorname{argmax} \varphi_p(d), d \in [0, d_{max}]. \tag{11}$$

The proposed refinement improves the matching accuracy. First, it acts as a regularization in local support regions and the regularization effectively removes the spurious errors. Second, by propagating the disparity information of their neighbors, the refinement infers the disparity of occluded areas and often attains accurate results for the occluded pixels.

1.5.3 Small "hole" filling

"Small hole" filling: According to our observation, remaining artifacts are mainly composed of some "small holes" and outliers around object boundary, as shown in Figure 5[2]. "Small holes" are dark regions with disparities much smaller than their neighbors. They can be detected by comparing disparities with their neighbors. After detecting hole-pixel, we use the most appropriate disparity in its neighborhood (both horizontal and vertical) to update it. Commonly, the hole-pixel p is updated by the pixel with smaller

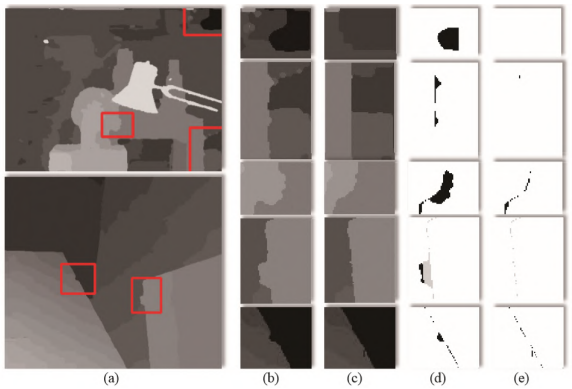


Figure 5: "Remaining artifacts" before and after RADAR, the first row represents "small hole"; the 2nd to 4th rows represent convex regions; the last row represents concave region. (a) Disparity maps before RADAR. (b) Close-up of rectangles in (a). (c) Results after RADAR. (d) Error maps of (b). (e) Error maps of (c)

disparity (background pixel), but if the pixel on the smaller disparity side of p is also invalid (hole-pixel), it should be updated by the pixel on the other side, as shown in follows,

$$d_p^* = \begin{cases} \max\{d_p^{'}, d_p^{"}\} & d_p^{'} \cdot d_p^{"} \leq d_{tres}^2 \\ min\{d_p^{'}, d_p^{"}\} & d_p^{'} \cdot d_p^{"} \geq d_{tres}^2 \end{cases} \tag{12}$$

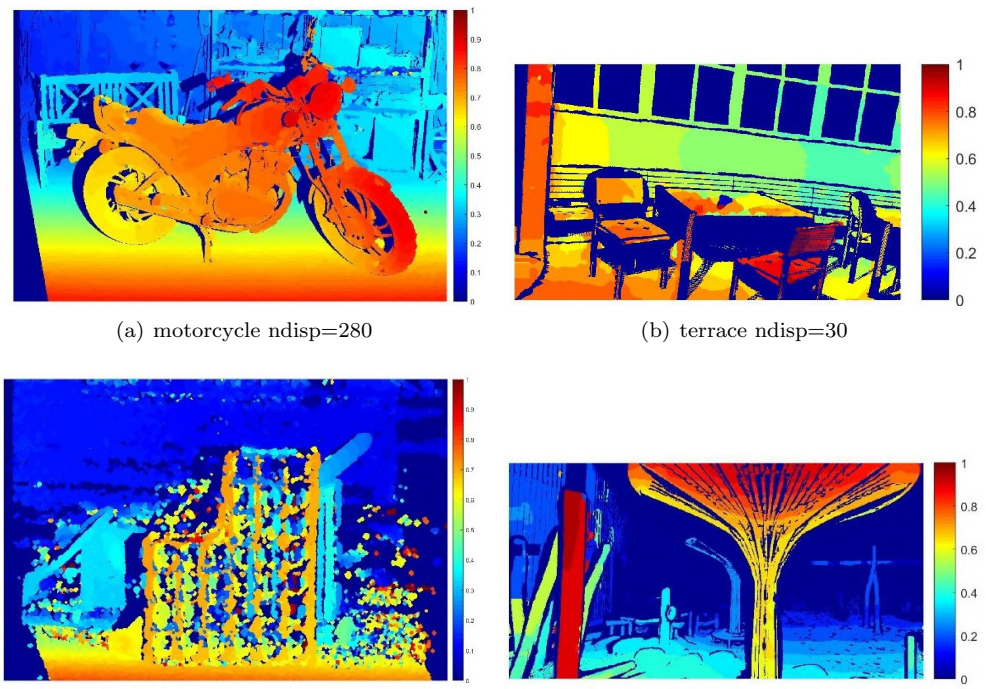
$$d_{tres} = \rho * d_{max} \tag{13}$$

Where $d_p^{'}$ and $d_p^{"}$ are the nearest taking oneone direction as an example pixels' disparities larger than d_{tres}, and d_{max} is the maximum disparity, while ρ is an empirical penalty of 1/7. The updated disparity is denoted as d_p^*.

2 Results

Test set	Motorcycle	Terrace	Playground	Sword
PSNR of masked disparity map[1](dB)	25.12	28.18 (50.28)	25.92 (48.92)	13.70
PSNR of unmasked disparity map(dB)	20.05	7.28(29.38)	11.01(34.01)	12.30
Total time consumption(s)	1231	29.25	42.93	1596

1: Masking indicates removing the invalid points in the ground truth from the disparity map.
The time consumption is available in the following report.
Test environment: CPU:
Intel i7-6700H@2.60GHz
RAM:16GB
PS: All the disparity maps are normalized with ndisp.
Our PSNR is calculated using the peak value which is the maximum value of the Ground Truth.
The value of PSNR in the second column and third column in the bracket represents when the peakvalue is 255.



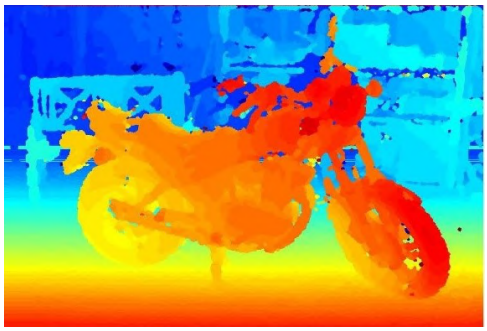
(a) motorcycle ndisp=280

(b) terrace ndisp=30

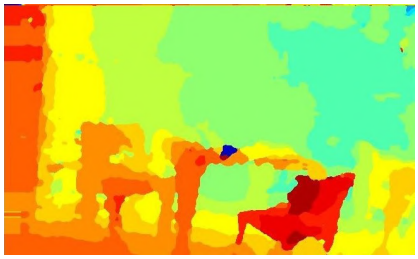
(c) sword ndisp=370

(d) playground ndisp=30

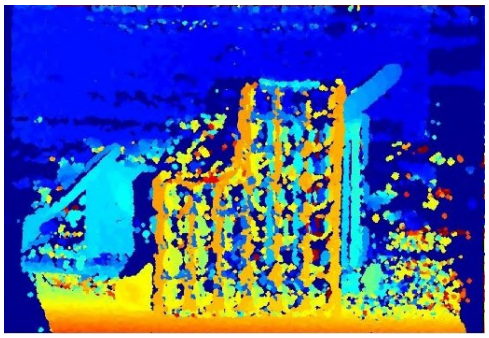
Figure 6: Masked disparity maps



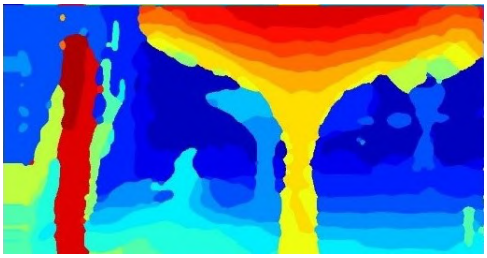
(a) motorcycle



(b) terrace



(c) sword

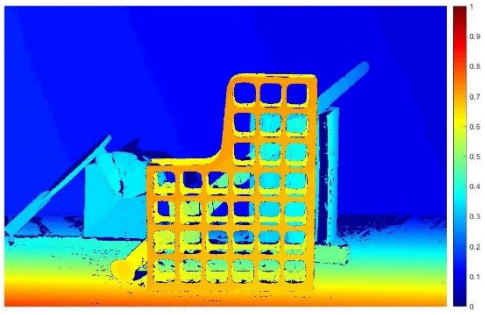


(d) playground

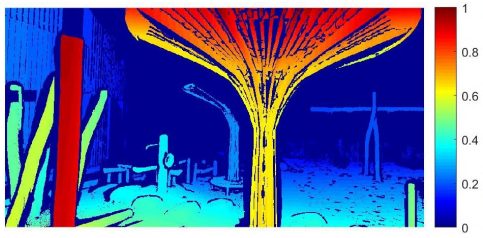
Figure 7: Unmasked disparity maps

(a) motorcycle

(b) terrace



(c) sword



(d) playground

Figure 8: Ground Truth

2.1 Test set: Motorcycle

Performance:

Test set	Motorcycle
PSNR of masked disparity map(dB)	25.12
PSNR of unmasked disparity map(dB)	20.05

Time consumption analysis:

Census Transform	14s
Window Matching	1154s
Refinement	63s
Total:	1231s

Parameter setting:

Algorithm	Window matching with fixed window size and joint cost function
Maximum disparity range	280
Window radius	12 pixels
Refinement	
Prior majority vote	2 iterations with window radius of 6
Hole_filling:3*r4	3 iterations with window radius of 4
Posterior majority vote	2 iterations with window radius of 6
Background filling	Yes

The mask file of motorcycle test set is acquired from http://vision.middlebury.edu/stereo/data/2014/. The mask file indicates the invalid points in the ground truth.
After removing the invalid points in the ground truth, the following disparity map is acquired.

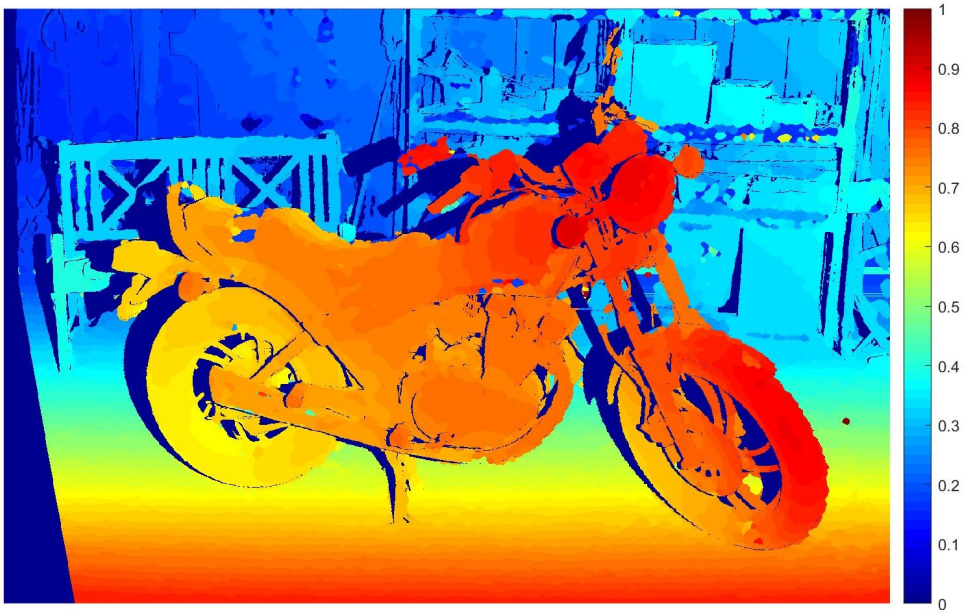


Figure 9: masked disparity map motorcycle

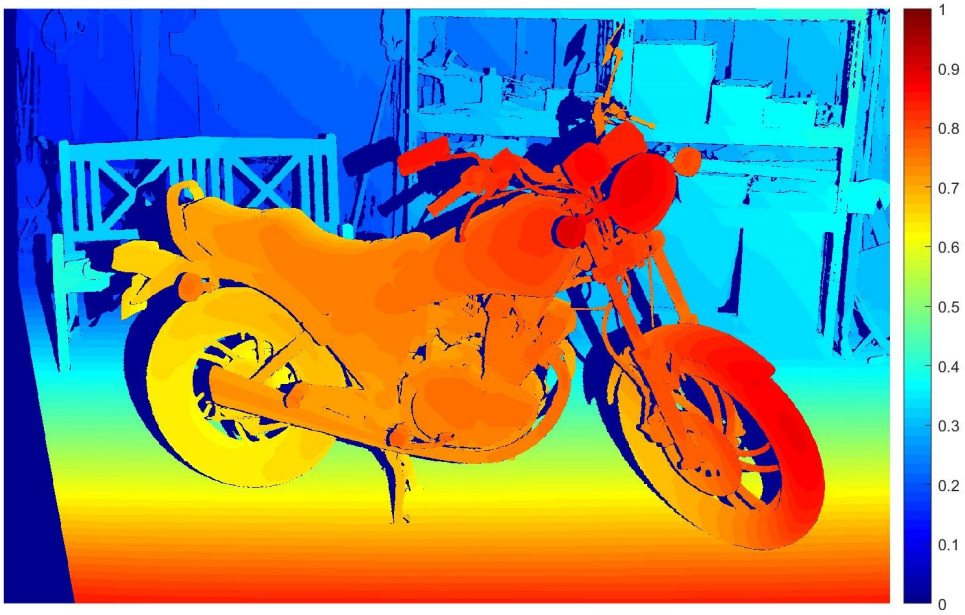


Figure 10: masked ground truth

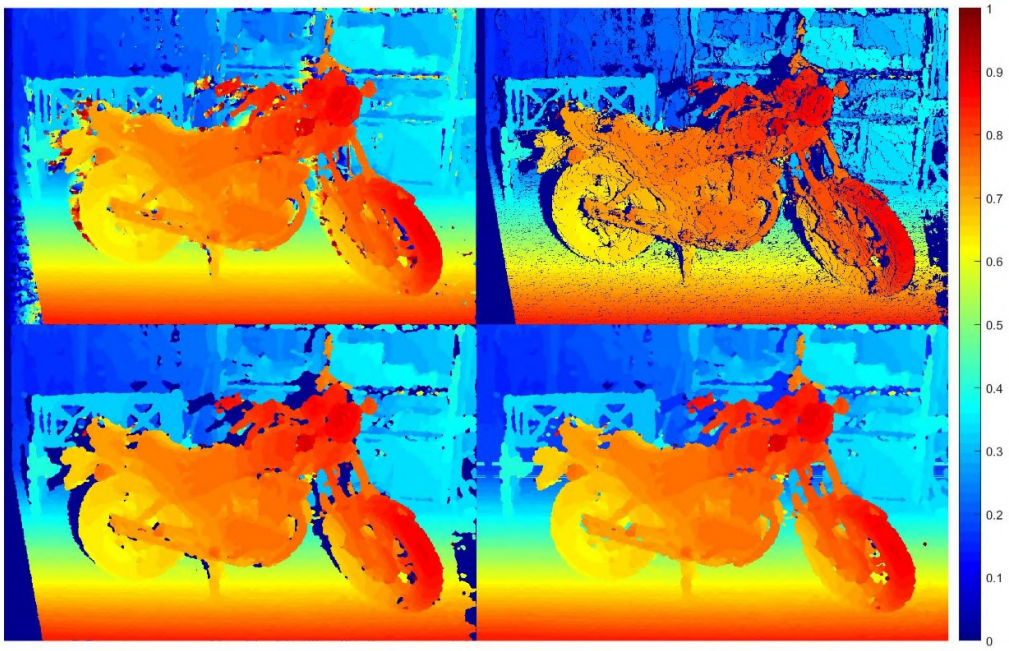


Figure 11: left upper :Original disparity map before refinement, left lower:Sparse disparity map after majority vote and hole filling, right upper:Sparse disparity map after consistency check, right lower:Dense disparity map after background filling

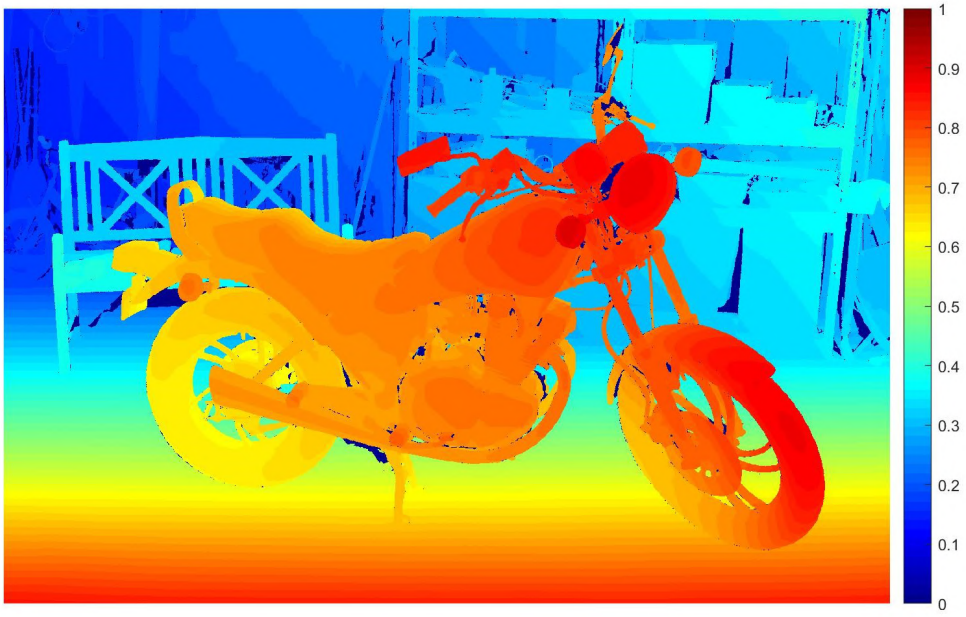


Figure 12: Ground truth motorcycle

2.2 Test set: Terrace

Performance:

Test set	Terrace
PSNR of masked disparity map(dB)	28.18
PSNR of unmasked disparity map(dB)	7.28

Time consumption analysis:

Census Transform	0.64s
Window Matching	25.13s
Refinement	3.48s
Total:	29.25s

Parameter setting:

Algorithm	Window matching with adaptive window size and cost function based on Hamming distance
Maximum disparity range	30
Adaptive window radius range:	2-23 pixels
Penalty coefficient	Beta =25, Gamma=2
Refinement	
Prior majority vote	2 iterations with window radius of 3
Hole_filling:3*r4	3 iterations with window radius of 4
Posterior majority vote	2 iterations with window radius of 3
Background filling	Yes

Only a masked ground truth is provided, after removing the invalid points in the ground truth, the following disparity map is acquired.

Figure 13: masked disparity map terrace

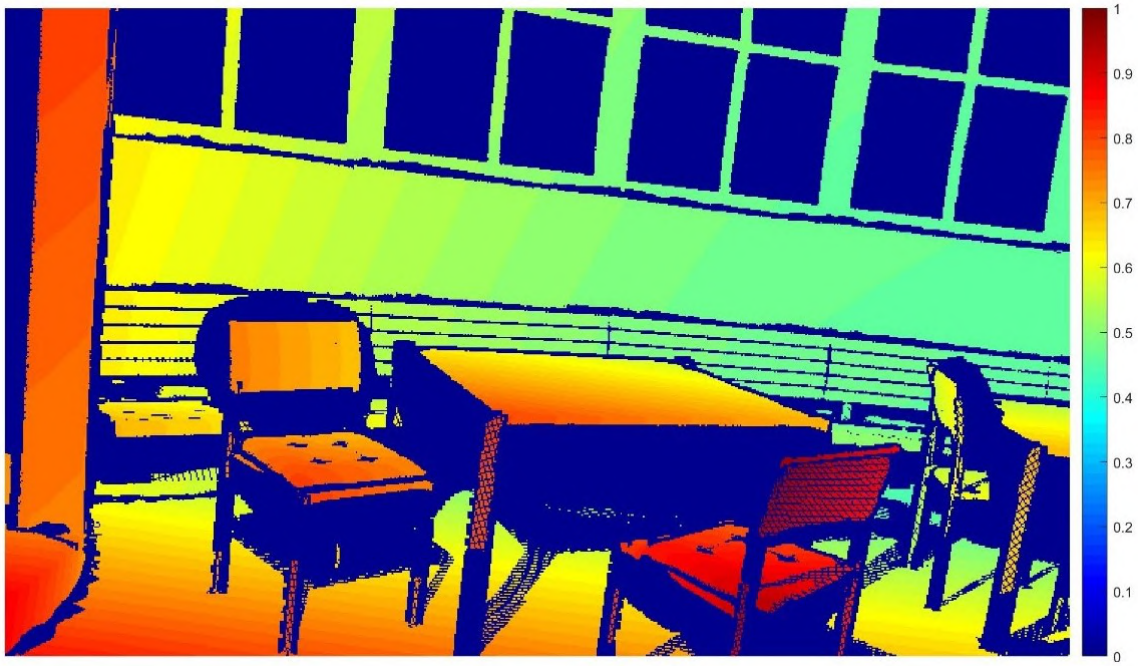


Figure 14: masked ground truth

Figure 15: left upper :Original disparity map before refinement, left lower:Sparse disparity map after majority vote and hole filling, right upper:Sparse disparity map after consistency check, right lower:Sparse disparity map after background filling

2.3 Test set: Playground

Performance:

Test set	Playground
PSNR of masked disparity map(dB)	25.92
PSNR of unmasked disparity map(dB)	11.01

Time consumption analysis:

Census Transform	1.01s
Window Matching	36.63s
Refinement	5.29s
Total:	42.93s

Parameter setting:

Algorithm	Window matching with adaptive window size and cost function based on Hamming distance
Maximum disparity range	30
Adaptive window radius range:	2-35 pixels
Penalty coefficient	Beta =30, Gamma=2
Refinement	
Prior majority vote	2 iterations with window radius of 3
Hole_filling:3*r4	3 iterations with window radius of 4
Posterior majority vote	2 iterations with window radius of 3
Background filling	Yes

Only a masked ground truth is provided, after removing the invalid points in the ground truth, the following disparity map is acquired.

Figure 16: masked disparity map playground

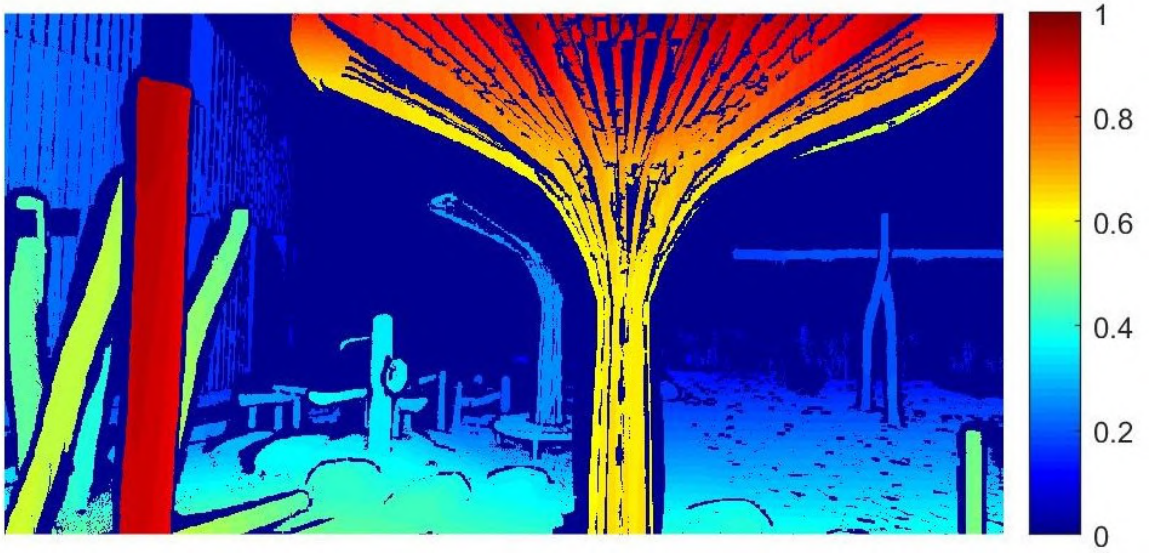


Figure 17: masked ground truth

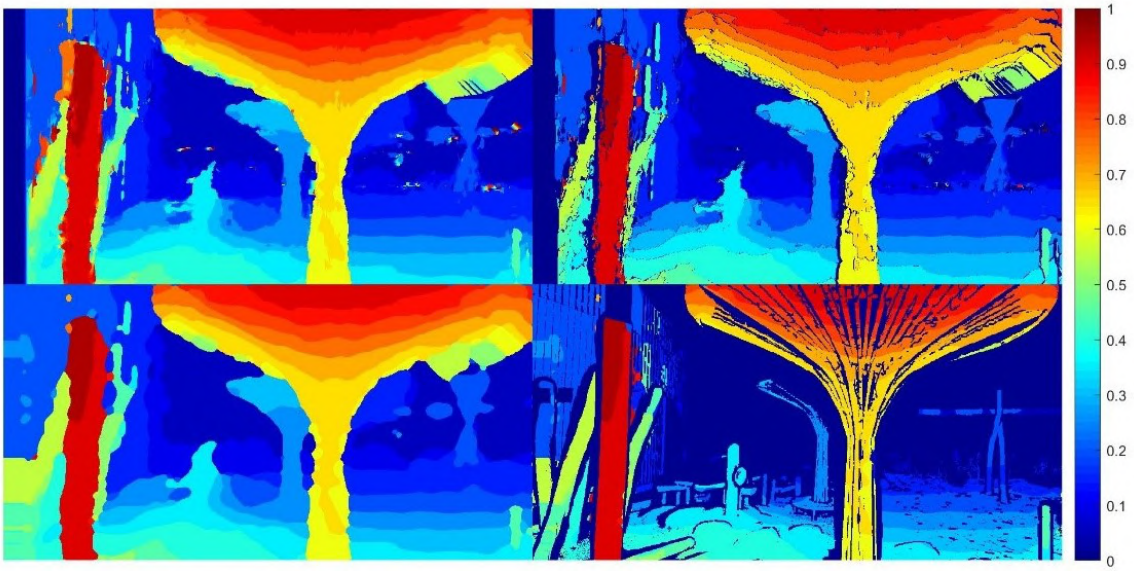


Figure 18: left upper :Original disparity map before refinement, left lower: Disparity map after majority vote, hole filling and background filling , right upper:Sparse disparity map after consistency check, right lower:Sparse disparity map after masking

2.4 Test set: Sword

Performance:

Test set	Sword
PSNR of masked disparity map(dB)	13.70
PSNR of unmasked disparity map(dB)	12.30

Time consumption analysis:

Census Transform	12.8s
Window Matching	1420s
Refinement	163s
Total:	1595.8s/26.6min

Parameter setting:

Algorithm	Window matching with fixed window size and joint cost function
Maximum disparity range	370
Window radius:	15
Penalty coefficient	Beta =30, Gamma=2
Refinement	
Prior majority vote	2 iterations with window radius of 3
Hole_filling:3*r4	3 iterations with window radius of 4
Posterior majority vote	2 iterations with window radius of 3
Background filling	Yes

Only a masked ground truth is provided, after removing the invalid points in the ground truth, the following disparity map is acquired.

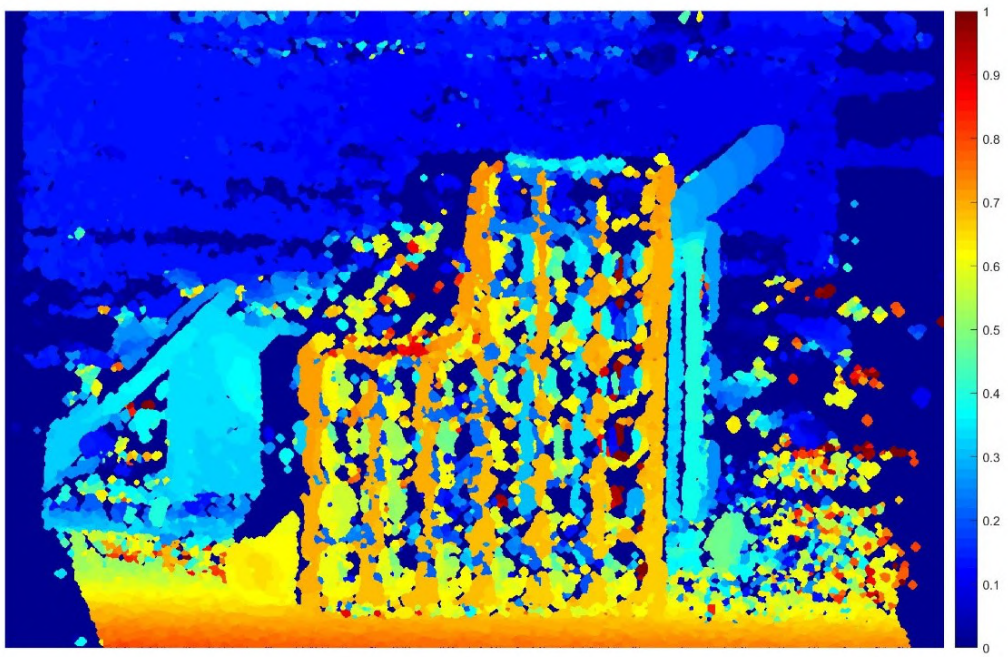


Figure 19: masked disparity map sword

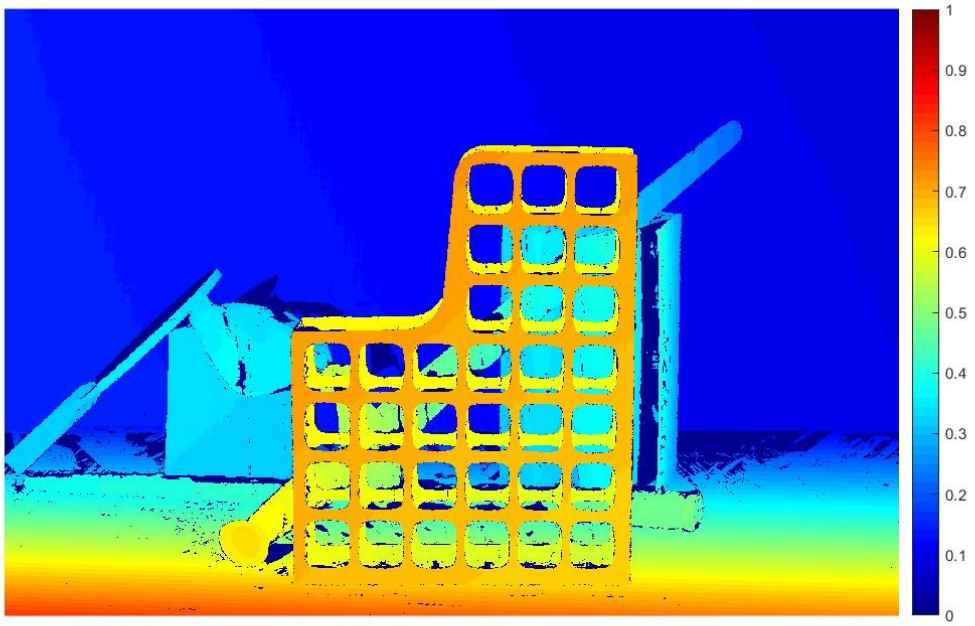


Figure 20: masked ground truth

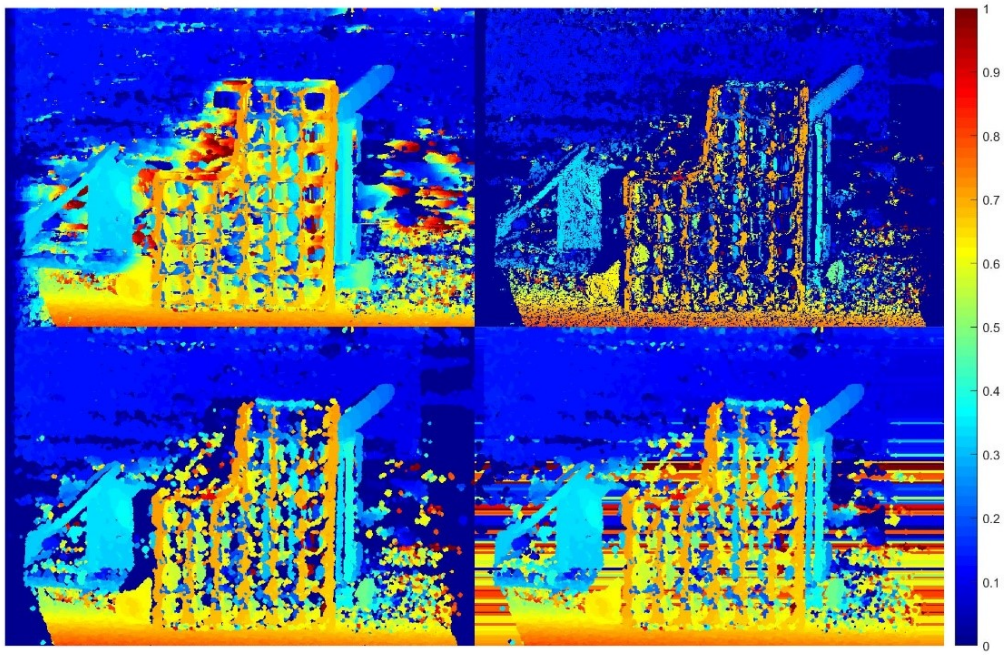


Figure 21: left upper :Original disparity map before refinement, left lower:Sparse disparity map after majority vote and hole filling, right upper:Sparse disparity map after consistency check, right lower:Dense disparity map after background filling

3 GUI

The GUI part is used to display an interface that helps to operate the total program. With this part, people can easily give the path of the images that they want to process. After clicking the button with different functions, we can get the results shown in the interface.

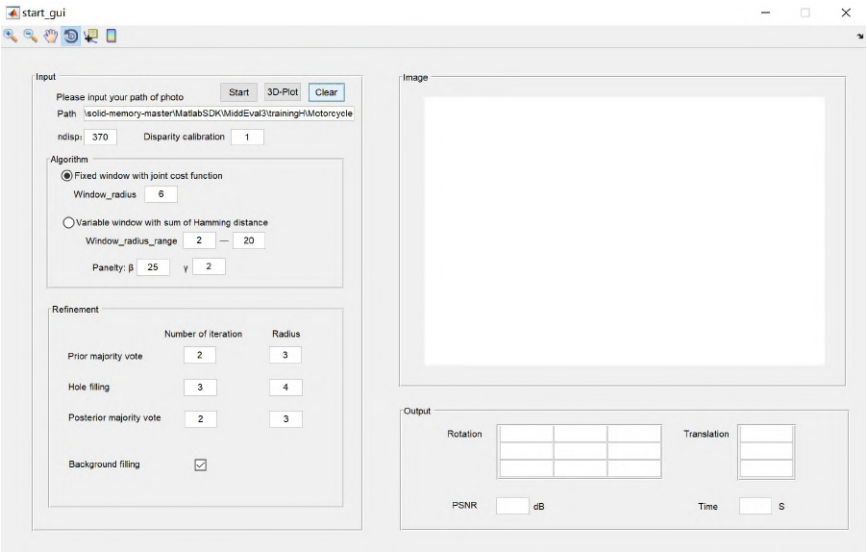


Figure 22: Original GUI interface

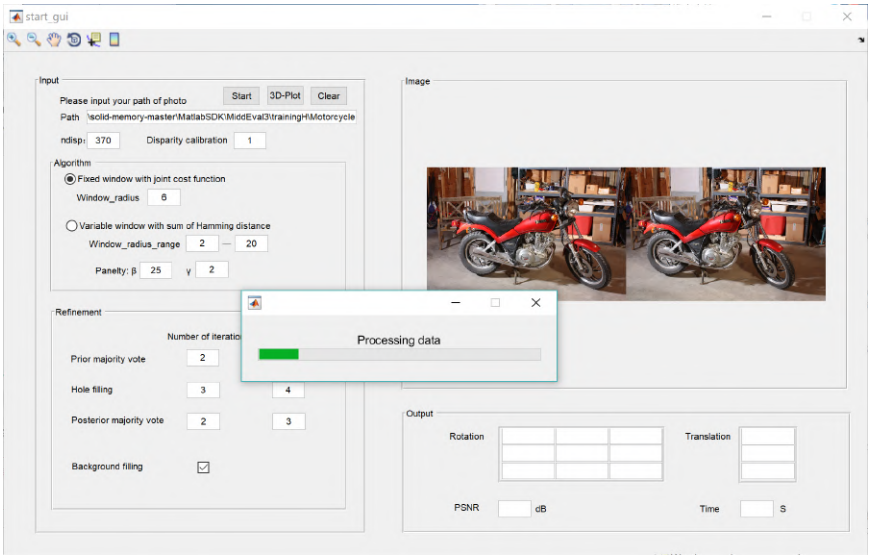


Figure 23: Load images in the GUI

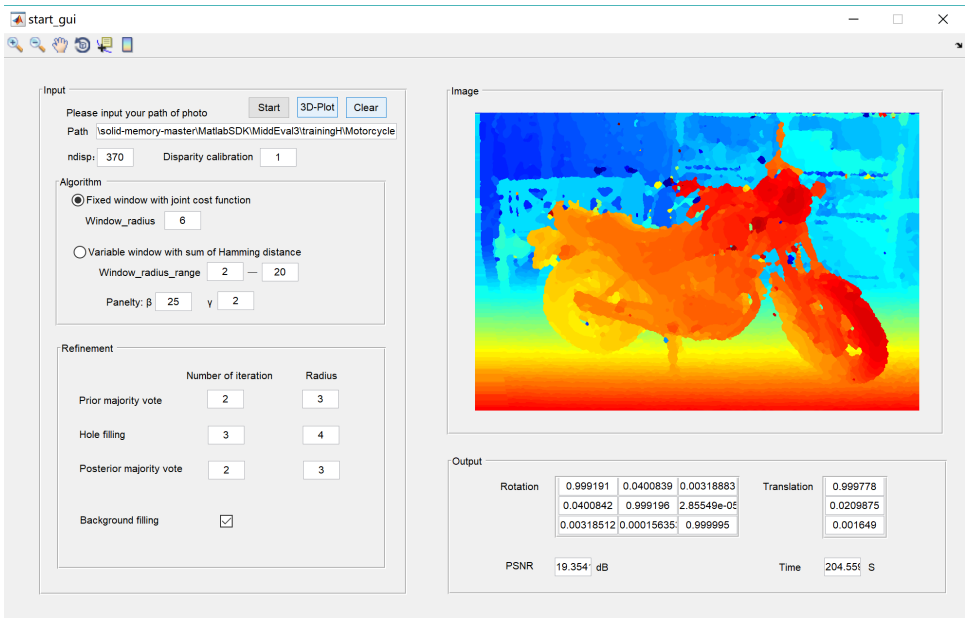


Figure 24: Result disparitymap

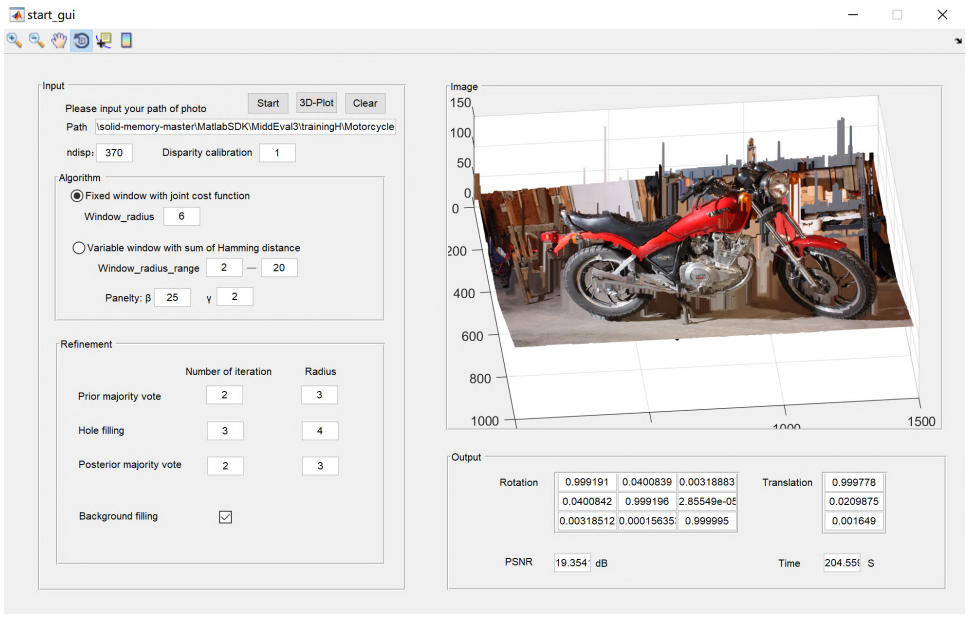


Figure 25: Result 3D plot

References

[1] Andrea Fusiello, Vito Roberto, and Emanuele Trucco. Efficient stereo with multiple windowing. In *cvpr*, volume 97, page 858. Citeseer, 1997.

[2] Jianbo Jiao, Ronggang Wang, Wenmin Wang, Shengfu Dong, Zhenyu Wang, and Wen Gao. Local stereo matching with improved matching cost and disparity refinement. *IEEE MultiMedia*, 21(4):16–27, 2014.

[3] Jaeryun Ko and Yo-Sung Ho. Stereo matching using census transform of adaptive window sizes with gradient images. In *2016 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, pages 1–4. IEEE, 2016.

[4] Olga Veksler. Fast variable window for stereo correspondence using integral images. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, volume 1, pages I–I. IEEE, 2003.

[5] Paul Viola, Michael Jones, et al. Rapid object detection using a boosted cascade of simple features. *CVPR (1)*, 1(511-518):3, 2001.

[6] Xiongren Xiao, Zheng Tian, Cheng Xu, Zhiqi Li, and Xiaodong Wang. Fast stereo matching using adaptive window based disparity refinement. *Journal ISSN*, 2368:5956, 2016.

[7] Ke Zhang, Jiangbo Lu, Qiong Yang, Gauthier Lafruit, Rudy Lauwereins, and Luc Van Gool. Real-time and accurate stereo: A scalable approach with bitwise fast voting on cuda. *IEEE Transactions on Circuits and Systems for Video Technology*, 21(7):867–878, 2011.