Robust and accurate depth estimation by fusing LiDAR and stereo

Guangyao Xu, Xuewei Cao, Jiaxin Liu, Junfeng Fan, En Li* and Xiaoyu Long

1 Institute of Automation, Chinese Academy of Sciences, Beijing 100000, People’s Republic of China
2 State Grid Liaoning Electric Power Company Limited, Shenyang 110000, People’s Republic of China

E-mail: en.li@ia.ac.cn

Received 10 April 2023, revised 25 June 2023
Accepted for publication 11 August 2023
Published 22 August 2023

Abstract
Depth estimation is one of the key technologies in some fields such as autonomous driving and robot navigation. However, the traditional method of using a single sensor is inevitably limited by the sensor’s performance. Therefore, a precise and robust method for fusing LiDAR and stereo cameras is proposed. This method fully combines the advantages of the LiDAR and stereo cameras, which can retain the advantages of the high precision of the LiDAR and the high resolution of images respectively. Compared with the traditional stereo matching method, the texture of the object and lighting conditions have less influence on the algorithm. Firstly, the depth of the LiDAR data is converted to the disparity of the stereo camera. Because the density of the LiDAR data is relatively sparse on the y-axis, the converted disparity map is up-sampled using the interpolation method. Secondly, in order to make full use of the precise disparity map, the disparity map and stereo-matching are fused to propagate the accurate disparity. Finally, the disparity map is converted to the depth map. Moreover, the converted disparity map can also increase the speed of the algorithm. We evaluate the proposed pipeline on the KITTI benchmark. The experiment demonstrates that our algorithm has higher accuracy than several classic methods.

Keywords: stereo matching, LiDAR, depth estimation, multi-sensor fusion, up-sampling

1. Introduction
The stereo camera is susceptible to interference from external environmental factors, such as light conditions. It can get more information from the other view, which do not improve the cameras’ anti-interference ability. The essential reason is that stereo matching algorithm need to find the same pixel between different images, but the solid color objects and the noise of images both affect this procession. Contrary to cameras, LiDARs have high anti-interference performance, which can obtain accurate depth information in complex environments, but the data of LiDARs is dense on the x-axis, but the low resolution on the y-axis. In addition to the stereo camera and LiDAR, RGB-D cameras are also often used to measure depth information, such as Kinect [1] and Realsense [2], which can quickly and accurately get depth information. They usually consist of a projector and a camera or two. The different RGB-D cameras would project different pattern, which may result in difference performance. Moreover, the RGB-D camera is generally suitable for measuring short-distance targets, because patterns would become blurry in the distance and difficult to capture clearly. And sunlight can also interfere with its structured light projector, so it is usually difficult to be used in the outdoor environment. In general, each sensor has its own strengths and limitations. The strategy of multi-sensor fusion is to make full use of the advantages of different sensors and overcome their limitations. At present, most of the fusion works of stereo and LiDAR are to integrate the data of the stereo camera and the 3D LiDAR in different ways.

* Author to whom any correspondence should be addressed.
to achieve an accurate estimation of the depth [3–8]. Although recently scholars have proposed many new stereo and LiDAR fusion methods and achieved good results, the new methods almost used deep learning methods which use massive data. But the lack of experimental data and the limitations of computing equipment prompted us to study new stereo camera and LiDAR fusion methods.

In the process of fusing the stereo camera and LiDAR, the positional relationship between the LiDAR and stereo camera is required. It builds a bridge between different sensors to facilitate the acquisition of the correspondence between point clouds and images. According to statistics, after the LiDAR point clouds are projected onto the pixel plane, they only occupy about 6.8% [9] of the image pixel position. In addition, the depth information of the LiDAR point cloud also can be converted into the accurate disparity information of the stereo camera. If the disparity converted from the point clouds is not on the pixel plane of the stereo camera, it would be discarded. And the LiDAR point clouds can help us to reduce the possible disparity search range, thereby reducing the time required for the calculation. In order to make full use of this precise disparity information, it is upsamled. Then the precise disparity can be combined with stereo matching to find a suitable position using propagation.

The main contributions of our paper are: firstly, a new method of removing outliers is proposed. Secondly, a new pipeline of fusion of the stereo images and raw LiDAR data is proposed.

An experimental platform was built to verify the feasibility of our method. And we also conducted lots of experiments on the KITTI benchmark to evaluate the efficiency of the proposed method. Figure 1 illustrates the entire experimental process.

2. Related work

In order to obtain reliable depth information, many methods have been proposed. Recently, these methods have mainly focused on three aspects: stereo matching of stereo images, fusion of stereo camera and LiDAR data, and completion of depth maps.

2.1. Stereo matching

The patch match stereo (PMS) [10] uses the slanted support windows to calculate the disparity. In comparison to the traditional front-parallel windows, they can be easily fit the slanted disparity surface. This means that the slanted windows can get more precise than the front-parallel windows. But they also consume much more time. MC-CNN [11] is one of the earliest methods to use deep learning for stereo matching, showing outstanding performance on the KITTI benchmark and proving the good performance of deep learning neural networks in the field of stereo matching. Garg et al [12] creatively used an unsupervised deep learning method to estimate depth, because the ground truth is difficult to obtain. This method uses the photometric error as the loss function. But the results of this algorithm are a little fuzzy. Next, Godard et al [13] added left-right consistency to the unsupervised approaches. Compared with existing unsupervised ways, their method effectively improves robustness and performance. Deep learning has made great progress in the field of computer vision and is consistently at the top of datasets such as the KITTI benchmark [14]. However, the dependence of supervised learning methods on data and computing power can make it difficult to apply them. In comparison to supervised learning methods, unsupervised learning methods [15] do not rely on ground truth, making them easier to apply.

2.2. Monocular depth estimation

Many new methods have been proposed with monocular camera recently. In contrast with stereo cameras, the monocular camera has low cost and simple structures, which make them easy to deploy. Monocular cameras typically obtain depth information of the captured object through motion using structure from motion [16]. Bartoccioni et al [17] proposed a monocular camera and simple LiDAR fusion method. The LiDAR is inexpensive and only contains four scan lines. The LiDAR data is added to solve the scale-ambiguity and infinite-depth issues. They fuse two neural networks, self-supervised image depth estimation [18] and fully-supervised depth completion [19]. Sparse LiDAR data replaces the dense LiDAR data and the ground truth as inputs and supervised data, respectively. This excessively relies on the four scanning lines to calculate the depth map and decrease the precision of the predicted.
depth map, which is input data to calculate the photometric loss. This may cause overfitting of sparse LiDAR data. Vitor et al [20] proposed a new neural network called DepthFormer, which enhances multi-view feature matching performance by combining cross-attention and self-attention mechanisms with depth discretized epipolar sampling. The DepthFormer is more robust than the method of calculating the depth value using photometric loss, as the feature is more resistant to interference than colors. However, since the features are still composed of pixels’ different colors, they are also susceptible to environmental interference. Xingyu et al [21] designed a new neural network to solve the edge-fattening issue. During matching process, some pixel cannot detect its corresponding pixels due of occlusions. However, the photometric loss function searches for a position with small losses, which generally falls onto the edge of the object that occludes the pixel. This creates the edge-fattening problem. They added the semantic map as input, which utilized its distinct boundaries to provide precise boundary information to the neural network. However, this causes algorithm accuracy to depend on the accuracy of the input semantic map that also increases computation time.

2.3. LiDAR and stereo fusion

In the field of multi-sensor fusion, several methods have also been proposed. In [3], Shivakumar et al performed anisotropic diffusion of depth data in the cost space of the SGM method. This method can be run in real-time on the TX2 platform. In [4], Will Maddern and Paul Newman proposed the use of probabilistic methods for fusing LiDAR and stereo cameras. However, the performance of this method may degrade in areas where LiDAR data is missing. Park et al [22] used deep learning methods for depth estimation with LiDAR and stereo camera, when comparing with traditional methods. They designed a two-step cascade deep architecture consisting of a disparity fusion module and a disparity refinement module. The features were extracted from LiDAR data and stereo disparity, which were fused in the LiDAR-Stereo fusion module. The fused disparity and RGB image were then used to estimate the real depth data in the refinement network. This method uses stereo disparity maps directly, but they can be refined by LiDAR data in the stereo matching process. The input data should be as accurate as possible for optimal performance. Wang et al [6] introduced Input fusion and CCVNorm into the stereo and LiDAR fusion network, achieving good results. Input fusion combines the LiDAR pair and the image pair as a fusion layer to learn the joint feature representation. CCVNorm adjusts the cost volume features based on LiDAR data. Choe et al [7] proposed a network that includes feature extraction, volume aggregation, and depth regression stages. The volume aggregation stage combines these features and calculates the geometric cost. The depth map is calculated in the depth regression stage. While the methods proposed by [6, 7, 22] require extracting features from LiDAR data, it may not contain sufficient information. Our proposed approach leverages the benefits of LiDAR data to improve stereo matching results without any special requirements for LiDAR information.

2.4. Depth completion

Some scholars propose to directly use raw LiDAR data for depth completion. We mainly focused on depth completion using traditional methods. Huber and Kanade [23] propose the depth map up-sampling method. The general process of this approach is to first sample in the horizontal direction and then sample in the vertical direction. Premebida et al [3] used bilateral filtering for up-sampling. In figure 4(a), moreover, the LiDAR data of different depths are projected onto the same region, especially in the edge area. In order to eliminate the error points, they used DBSCAN [24] method to cluster the LiDAR data based on different depths and then retain the data on the correct depth layer. But this method is very time-consuming. And the depth of LiDAR data is smoothed, which would change data characteristics. Ku et al [25] performed morphological processing on the depth image and achieved good up-sampling results. This method runs in real-time. However this method has the same disadvantage as in [3]. In addition to the above-mentioned, Hengjie et al [26] proposed a deep learning method for depth estimation using single-line LiDAR and an image. Compared with multi-line LiDAR, the single-line LiDAR has the advantages of simple structure and low price. But this is an unsolvable problem in math.

The rest of the paper is organized as follows: section 3 proposes the disparity-map up-sampling method. In section 4, the detailed process of depth estimation is introduced. Section 5 presents some experiments and data analysis. Section 6 provides a summary and improvement measures.

3. Disparity-map upsampling formulation

To fuse LiDAR data and stereo images, the LiDAR depth is converted into the disparity and then upsampled. The disparity map upsampled and depth completion are treated as the same tasks in this paper. The pair of stereo images is divided into left image L and right image R. Additionally, the LiDAR data corresponding to the stereo images are projected onto the image plane of L denoted by \( \Pi_L \) or the image plane of R denoted by \( \Pi_R \) as the disparity map according to equations (1) and (4). The stereo camera is a parallel optical axis system,

\[
Y = K[R|t]X
\]  

(1)

where \( X = (x', y', z', 1)^T \) denotes a 3D LiDAR point. And \( Y = (x, y, 1)^T \) denotes a pixel in the camera image, the position \((x, y)\) of which may not be an integer pixel coordinate. So it can be rounded,

\[
K = \begin{bmatrix}
f_x & 0 & c_x & 0 \\
0 & f_y & c_y & 0 \\
0 & 0 & 1 & 0
\end{bmatrix}.
\]  

(2)

\( K \) is the camera projection matrix. \((f_x, f_y)\) and \((c_x, c_y)\) are the focal length and principal point of stereo cameras. \( R \in \mathbb{R}^{3 \times 3} \) and \( t \in \mathbb{R}^{3 \times 1} \) are rotation and translation matrices from LiDAR to camera.
regions with a greater depth compared to the red points. This phenomenon is explained in the [3].

This problem causes great trouble for our disparity map up-sampling. To a certain extent, algorithm 1 can be used to solve this problem by clearing the outliers between scan lines. A brief summary of algorithm 1 is as follows: the disparity from LiDAR data has a noticeable feature: high resolution in the horizontal direction and low resolution in the vertical direction. The first step is to complete these scan lines in the horizontal direction, as shown in figure 4(b). This is a straightforward procedure in which, if the adjacent left and right location of the LiDAR disparity is zero, the LiDAR disparity is assigned to them. After that, the disparity coordinates (x, y) of Π are traversed vertically to find the disparity coordinates (x, y_up) and (x, y_down), as depicted in figure 3. If the absolute difference between Π(x, y) and either Π(x, y_up) or Π(x, y_down) exceeds a threshold value, such as Maxdisp or Maxcolor, the disparity value will be assigned a value of 0. This means that outliers are eliminated. If the input images are color images, then color can also serve as a parameter. Figure 4(c) is the result of removing error points. And then the y-axis of figure 4(c) is linear interpolated between scan lines according to equation (5).

\[
\Pi(x, y) = \frac{y - y_{up}}{y_{down} - y_{up}} \Pi(x, y_{down}) + \frac{y_{down} - y}{y_{down} - y_{up}} \Pi(x, y_{up})
\]

where Π(x, y) is the disparity at (x, y). y_up and y_down are y coordinates that are the closest to (x, y) and are distributed on the scan lines. Figure 5(a) shows the result after linear interpolation. Due to the fact that linear interpolation is not good at preserving edges, some other scholars like [3] use bilateral filtering to up-sample the disparity map. In [3], disparity map is up-sampled by bilateral filtering as shown in the figure 5(b).

\(BF^\ast\) represents the disparity map using the up-sampling method proposed in this paper, and \(BF^\ast\) represents the up-sampling method proposed in [3]. Compared with \(BF^\ast\) [3], our method \(BF^\ast\) has better accuracy and robustness. In section 5, we compared the two up-sampling methods in detail.

4. Depth estimation

The most important step of depth estimation is the cost calculation. However, due to environmental factors, the

\[
[R|t] = \begin{bmatrix}
R & t \\
0 & 1
\end{bmatrix}
\] (3)

And then the depth \(z'\) of \(X\) is converted to disparity \(d\) at the pixel coordination \((x, y)\).

\[
d = \frac{BF}{z'}
\] (4)

Algorithm 1. Eliminate the overlapping points.

1. Get projection disparity map
2. Complement the scan line by interpolation method
3. UP_FLAG ← false
4. DOWN_FLAG ← false
5. for the coordinate \((x_d, y_d)\) of each \(d\) in \(Π\) do
6. for \(Y_{up} ← y_d\) to \(y_{up}\) do
7. if \(Y_{up} > 0\) and \(Π(x, Y_{up}) - Π(x_d, y_d) > Maxdisp\) and \(Color(x, Y_{up}) - Color(x_d, y_d) < Maxcolor\) then
8. UP_FLAG ← true
9. end if
10. end if
11. for \(Y_{down} ← y_d\) to \(y_{down}\) do
12. if \(Y_{down} < MaxY\) and \(Π(x, Y_{down}) - Π(x_d, y_d) > Maxdisp\) and \(Color(x, Y_{down}) - Color(x_d, y_d) < Maxcolor\) then
13. DOWN_FLAG ← true
14. end if
15. end for
16. if UP_FLAG = True and DOWN_FLAG = True then
17. \(Π(x_d, y_d) = 0\)
18. end if
19. end for

where \(B\) and \(F\) are the baselines and the focal length of stereo cameras. Figure 2 shows the process of projection. In general, we uniformly refer to the left disparity map as Π, which represents the baseline disparity map. The disparity map Π can be considered an 8-bit image, with the disparity in Π denoted as \(d\). Although the points in Π are sparse, there are some points of different depths that are projected to the same local area, such as the magnified area in figure 4(a). The blue points represent
images captured by the camera often lose part of the information. Moreover, LiDAR shows strong robustness to environmental noise. We add LiDAR data and aim to obtain more accurate matching costs to propagate the disparity. This section contains three subsections: window model, matching costs, and minimizing matching costs. In the stereo matching process, using a single pixel to calculate matching costs can be unreliable and result in large errors, as it is susceptible to interference. The window aggregates several neighboring pixels to calculate the matching costs based on a central pixel. This approach can reduce noise and improve the precision of algorithms. The window model introduces the slanted support model, and matching costs can evaluate the quality of windows based on pixels. In general, the smaller the matching costs, the more precise the corresponding disparity. To achieve more accurate disparity, minimizing matching costs utilizes various methods to decrease the matching costs.

4.1. Window model

Front-parallel windows can be expressed as windows parallel to the image plane. The slanted support window [10] can be denoted as the window that fits the disparity surface. Moreover, the disparity of front-parallel windows is integer-based disparity, whereas the slanted support window is based on sub-pixel disparity. For instance, slanted support window $d$ of figure 6 is sub-pixel disparity between disparity 1 and 2. There is a significant difference between the slanted support window and the front-parallel window proposed as shown in figure 6. In section 5, we compare the effects of the two window models in detail. Slanted support window is used to allocate a plane $\xi$ with the least matching cost for each pixel $p$. The plane $\xi$ is determined by the point $(x, y, d)$ and a random unit normal vector $\vec{n} = (n_x, n_y, n_z)$ of the point. The plane $\xi_p$ can be represent as

$$d = a_\xi x + b_\xi y + c_\xi$$

where $(x, y, d)$ is a point of the disparity surface and the $(x, y)$ is pixel coordination in the disparity map. $a_\xi, b_\xi$ and $c_\xi$ are the three parameters of the plane $\xi$. These parameters are obtained by the following three formulas,

$$a_\xi = -\frac{n_x}{n_z}$$

$$b_\xi = -\frac{n_y}{n_z}$$

$$c_\xi = \frac{n_d}{n_z}$$

Figure 4. The process of eliminating outliers from the disparity map.

Figure 5. The results of up-sampling on the disparity map. (a) uses linear interpolation and (b) uses bilateral filtering of [3].

Figure 6. Slanted support window and front-parallel window. The y-axis of the parallax plane is set to a specific value, so the three-dimensional scene is projected onto the two-dimensional image.
where the point \((x_0, y_0, z_0)\) is a point in the slanted support window. There are infinitely many planes such as \(ξ\), which prevent us from finding all possible planes. To solve this, the plane can be found within a search range where the optimal solution is most likely to appear. The search range is set between the minimum and the maximum disparity of the whole image in PMS [10]. However, it can also be set between scan lines after adding LiDAR data. Since the disparity obtained from the LiDAR data is sufficiently accurate, the disparity \(d\) on the scan line can be regarded as the maximum or minimum disparity of the pixels between the scan lines. If there are no two scan lines or only one scan line, the search space is set between the minimum and maximum disparity of the whole image. Allocating a smaller search space for disparity can significantly reduce the calculation time. In addition, if the normal vector equals \((0, 0, 1)\), the slanted window can be transformed into the front-parallel window.

4.2. Matching costs

After determining the window model, the matching cost also needs to be determined. \(p(x, y)\) can be denoted as a pixel coordinate of disparity in the disparity map for convenience. The aggregation cost of points \((p, d)\) on plane \(ξ\) is

\[
C(p, ξ_p) = \sum_{(p', d') \in W_p} \varphi(p, p') \cdot \phi(p', d' - (aξx + bξy + cξz))
\]

(10)

where \(W_p\) is a square window centered on \(p\). \((p', d')\) is pixel coordinate and disparity in the window \(W_p\) of \(p\).

The first item of equation (10) is the function \(\varphi(p, p')\) that is used to solve the edge-fattening problem [27].

\[
\varphi(p, p') = e^{-\frac{||I_p - I_{p'}||}{δ}}
\]

(11)

where \(δ\) is a custom parameter that controls the proportion of pixels of different colors in the original image. In this paper, the value of \(δ\) is 10. \(||I_p - I_{p'}||\) calculates the Manhattan Distance between the color value of \(p\) and \(p'\) in RGB space. The value of \(\varphi(p, p')\) would be close to 1 if color is similar. Equation (12) calculates the matching cost between \(p'\) and \(q\) and appears as the last section of equation (10). \(q\) is expressed as \(q = (aξx + bξy + cξz)\), which represents the matching points of \(p'\) obtained by subtracting the disparity of the \(x\)-axis from \(p'\) in the other view.

\[
\phi(p', q) = \alpha \cdot \min(||I_p - I_{q}||, λ_{col}) + \beta \cdot \min(||\nabla I_p - \nabla I_{q}||, λ_{grad}) + \gamma \cdot \min(||\Pi_{p'} - d'||, λ_{disp})
\]

(12)

Here, \(||I_p - I_{q}||\) and \(||\nabla I_p - \nabla I_{q}||\) are the Manhattan Distance in RGB and gradient. \(||\Pi_{p'} - d'||\) is the absolute difference of the disparity. \(||\Pi_{p'} - d'||\) is disparity of \(p'\) in up-sampled disparity map. \(d'\) is the disparity of \(p'\) in plane \(ξ\). The color and gradient come from the original image and the disparity comes from the disparity map. Parameters \(λ_{col}, λ_{grad}\) and \(λ_{disp}\) are the truncation costs. In the occluded area of the image and the blank area in the disparity map, the truncation costs can limit the aggregation cost to a rational range and increase the robustness of the algorithm. The parameters \(λ_{col}, λ_{grad}\) and \(λ_{disp}\) can be set to 10, 2 and 2. In [28], Rhemann et al proposes the matching cost of color and gradient, but the addition of LiDAR data allows the algorithm to have higher accuracy and robustness,

\[
α + β + γ = 1.
\]

(13)

To prevent the final aggregation cost value from being too large, the sum of \(α\), \(β\), and \(γ\) is limited to 1. They can also be considered as the ratios of color, gradient, and disparity, respectively, in the matching cost. Compared to the disparity from LiDAR data, the gradient is less precise. Although adding the gradient can affect the accuracy of the algorithm and increase computation time, it can also improve the algorithm’s robustness when the disparity does not exist in certain areas of the images.

4.3. Minimize matching cost

This section is used to determine the parameters of the plane that minimize the matching cost. The algorithm can be divided into three steps: initialization, disparity propagation, and plane refinement. The algorithm performs disparity propagation and plane refinement once every iteration. In this paper, the algorithm performed three iterations. And in the following subsection, some details of the algorithm can be introduced.

4.3.1. Initialization. The plane model has been determined in the section of the window model. Then each pixel needs to be assigned a plane, but many plane parameters have not been determined such as \(x, y, d, n_x, n_y, n_z\). So each pixel can be assigned a random unit vector \(n_x, n_y, n_z\). The plane model is the front-parallel windows, the unit vector is \(0\). And if the \(p'(x, y)\) has disparity \(d'\) in the up-sampling disparity map, the disparity \(d'\) would be assigned to \(d\). If no such value exists, a random initial value between the maximum and minimum disparity of the whole image would be assigned to \(d\).

4.3.2. Disparity propagation. Each pixel has been assigned a plane, and then the plane with the least cost of aggregation on each pixel needs to be found. Because adjacent pixels are likely to have similar planes, the planes with a relatively low aggregation cost can be found by spreading the planes of adjacent pixels. There are many ways to spread, such as from left to right, from top to bottom, from top left to bottom right, and so on. The method used in this paper was
4.3.3. Plane refinement. To further reduce the aggregating cost of equation (10), the plane needs to be refined. The refined plane is between the two propagation processes, and it provides the new plane with a lower aggregation cost for the propagation process if it is possible. This is also the most time-consuming step of the algorithm. We make use of parallel processing methods to reduce calculation time of this sub-process. In this process, the plane $\xi$ need to be represented a point $(x, y, d)$ and a normal vector $(n_x, n_y, n_z)$. Two parameters $\Delta_d^{MAX}$ and $\Delta_n^{MAX}$ are defined as the maximum allowed range of the disparity $d$ and $n$. If the pixel is located between two LiDAR scan lines, $\Delta_d^{MAX} = |\Pi(x, y_{up}) - \Pi(x, y_{down})|$. If not, $\Delta_n^{MAX}$ would be the maximum disparity of the image. Similarly, $\Delta_n^{MAX}$ is also the maximum allowable range of the vector. $\Delta_n$ is randomly selected from $(-\Delta_n^{MAX}, \Delta_n^{MAX})$ to calculate $d = d + \Delta_d$. $\Delta_n$ is also a random value in the $(-\Delta_n^{MAX}, \Delta_n^{MAX})$ and $n' = \text{unit}((\vec{n} + \Delta_n))$, where $\text{unit}()$ is unitized. Now all parameters of a new plane $\xi'$ are determined. If $C(p, \xi') > C(p, \xi_p)$, $\xi'$ can be used instead of $\xi_p$. This is an iterative process. $\Delta_d^{MAX}$ is initialized to $\Delta_d^{MAX}/2$ and $\Delta_n^{MAX} = 1$. $\Delta_d^{MAX}$ is updated as $\Delta_d^{MAX}/2$ and $\Delta_n^{MAX} = \Delta_d^{MAX}/2$ every iteration. This can reduce our search range. If $\Delta_d^{MAX} < 0.1$, refining the plane $\xi_p$ is aborted. In the first iteration, the disparity changes may be large. In subsequent iterations, the disparity changes may gradually decrease. Once the refining of the plane is completed, the plane can then restart the disparity propagation process.

5. Experiment and analysis

5.1. Experiment setup and settings

Our experimental device consists of a stereo camera and 3D LiDAR, as shown in the right part of figure 8. The left part of figure 8 shows the installation method of the device. Stereo images and LiDAR point clouds can be captured by this device.

To better evaluate errors, we evaluate the performance of our algorithm on the KITTI depth benchmark [29]. The data we use comes from the KITTI Raw dataset [14], which provides raw data from different road environments. The experimental device used in KITTI consisted of Velodyne HDL-64E LiDAR and two FLIR color cameras. The KITTI depth benchmark contains 46k ground truth data. We use 1k samples from this dataset, which contain different scenes for testing, because the test set of the KITTI depth benchmark does not provide stereo images or ground truth data. We would also like to note that we test the area covered by the raw LiDAR data. To ensure the correctness and fairness of the evaluation process, we used the evaluation development kit, which was provided by the KITTI depth benchmark. The configurations of our computer are i7-8700 and 8GB of RAM.

To help readers better understand the characteristics of our methods, we compare them in five different ways, with results shown in figure 9 and tables I-IV. In these tables, ‘$>$ 2px’ means that errors greater than 2 pixels are counted, as well as ‘$>$ 3px’ and ‘$>$ 5px’. OURS(BI) and OURS(BF) represent the results of combining stereo images with BI and BF, respectively [3]. RMSE, MAE, iRMSE, and iMAE are abbreviations for root mean squared error, mean absolute error, inverse root mean squared error, and inverse mean absolute error, respectively [30]. RMSE and MAE indicate the performance of the algorithm at distant locations, while iRMSE and iMAE increase the weights of small depth values, enabling them to better represent the performance of the algorithm at close locations. Our experience data was collected using $\alpha$, $\beta$, and $\gamma$ values set to 0.1, 0, and 0.9, respectively.
5.2. Impact of window size

The patch size is an important parameter that has a significant impact on the accuracy and processing time of the algorithm. Figure 9 shows the relationship between the window size, MAE, and processing time. Due to the presence of outliers, using a single pixel for matching is susceptible to being influenced by outliers. However, using all pixels in a window for matching can eliminate the influence of outliers to some extent. The window size of PMS is set to 40, which yields the minimum error as illustrated in figure 9(a). In subsequent experiments, the patch size of PMS was set to 40, which is similar to traditional stereo matching algorithms and makes the algorithm more robust by increasing the window size appropriately. Compared to PMS, the error of our methods is less sensitive to window size. The addition of LiDAR data improves the robustness of the algorithm, with our methods achieving the smallest error as shown in figure 9(a) when the window size is approximately 10. Furthermore, increasing the window size also increases the processing time for all methods, as shown in figure 9(b), as larger window sizes entail more calculations. The calculation speed of OURS(BF*) is 3.55 times faster than that of PMS. At the beginning of the OURS(BF*) curve, BF contributes most to the processing time. Additionally, because the up-sampled disparity map of BF* is denser than that of BI*, as shown in figure 5, the calculation time of OURS(BF*) is somewhat shorter than that of OURS(BI*) towards the end of the curve for OURS(BF*).

5.3. Upsampling experiment on LiDAR disparity map

BF is a method based on bilateral filtering [31], which narrows the distribution range of the disparity. This approach may be less conducive to correctly propagating the disparity, as it may amplify some errors when the ground truth data is far from the mean value. On the other hand, BI is an interpolation method that works well in filling in gaps between two LiDAR scan lines when the LiDAR beam is relatively dense. Since the surface of an object is divided into many different scan lines, the idea of differential geometry is used to understand the relationship between the object’s surface and the LiDAR scanning line.

For the aforementioned reasons, we can observe from table 1 that the accuracy of BI is higher than that of BF when the error is relatively small. The pixel distribution of BF* is more concentrated, and a large portion of its error is concentrated in ‘>2px’. However, the error in ‘>5px’ is relatively low.

5.4. Slanted support window experiment

We used different up-sampling methods to estimate depth, such as BI* and BF from [3]. Table 2 compares the performance of our fusion methods with that of the PMS algorithm. After adding LiDAR data, algorithm performance was significantly improved. OURS(BI*) performed better than OURS(BF*) because evenly distributed disparities are more conducive to correct disparity propagation. Accordingly, the experimental results show the OURS(BI*) to be the best performing method. Figure 10 shows the results of the three

---

**Table 1.** Up-sampling experimental.

| Method | >2px | >3px | >5px |
|--------|------|------|------|
| BI*    | 4.52%| 3.73%| 3.27%|
| BF*    | 16.13%| 5.74%| 2.87%|

Note: Bold values indicates the best performance among the comparisons.

**Table 2.** Slanted support window experiment.

| Method  | RMSE | MAE  | iRMSE | iMAE  |
|---------|------|------|-------|-------|
| PMS     | 4776 | 1611 | 18.15 | 6.10  |
| OURS(BF*) | 1787 | 651  | 3.48  | 2.43  |
| OURS(BI*) | 1282 | 302  | 1.65  | 0.80  |

Note: Bold values indicates the best performance among the comparisons.
algorithms, and figures 10(e) and (k) (figures 10(f) and (l)) indicate that adding gradient information and increasing window size can enhance the robustness in areas where LiDAR data is missing. At the same time, this can increase computation time and reduce accuracy.

5.5. Front-parallel window experiment

The performance of the slanted support window is better than that of the front-parallel window. However, the performance improvement of our methods is not as significant as that of PMS which uses the slanted support window. Additionally, OURS(\(BF^*\)) has a very small difference between using the slanted support window and the front-parallel window. This is also caused by the up-sampling method of \(BF^*\) which makes the disparity map smoother. The results are shown in table 3.

5.6. Comparison of different methods

We compared our method with different existing methods. Most scholars currently use deep learning methods to study the fusion of LiDAR and stereo. On the one hand, supervised learning methods rely on ground truth to train the network, which is expensive and difficult to obtain for depth estimation. In contrast, our method is not limited by datasets and offers a viable alternative to supervised learning methods. On the other hand, our method outperforms the latest unsupervised learning method [8] in terms of accuracy. In situations where ground truth is not available to train the neural network, our method would be a good choice. Furthermore, our method has achieved superior results in close-range locations, as indicated by the inverse depth results (iMAE and iRMSE) in table 4. Although our method has made significant progress in time compared to PMS, achieving real-time performance remains challenging.

5.7. Experiments on our equipment

This subsection discusses some key details of our experiment process. Our experimental setup consisted of a Velodyne VLS-128 and a ZED stereo camera. We used the Autoware autonomous driving software [32] to calculate the positional relationship between the stereo camera and the LiDAR. This allowed the LiDAR points to be transferred to the camera coordinate system. The software method we used is not sensitive to small objects. For the camera and LiDAR fusion system to function properly, the two sensors must collect data simultaneously. Timestamps were used to mark LiDAR frames and stereo images. The Velodyne VLS-128 has 128 LiDAR scan lines, in contrast to the Velodyne HDL-64E used in the KITTI dataset that has only 64 scan lines. A high-performance LiDAR can provide more detail in the depth map, as shown in figure 11.
Table 4. Experimental results of different methods.

| Method                        | Modality   | RMSE(mm) | MAE(mm) | iRMSE(1/km) | iMAE(1/km) | Time(s) |
|-------------------------------|------------|----------|---------|-------------|------------|---------|
| Park et al [22]              | Stereo + LiDAR | 2021     | 500     | 3.39        | 1.39       | 0.04    |
| LiStereo(self-supervised) [8] | Stereo + LiDAR | 1278     | 326     | 3.83        | 1.32       | —       |
| LiStereo(supervised) [8]     | Stereo + LiDAR | 834      | 284     | 2.20        | 1.10       | —       |
| CCVN [6]                     | Stereo + LiDAR | 750      | 253     | 1.40        | 0.81       | 1.01    |
| VPN [7]                      | Stereo + LiDAR | 640      | 206     | 1.88        | 1.00       | —       |
| OURS(BI∗)                    | Stereo + LiDAR | 1282     | 302     | 1.65        | 0.80       | 6.00    |

Note: Bold values indicate the best performance among the comparisons.

Figure 11. The results on our equipment. (a)(c) The left images. (b)(d) The depth map.

6. Conclusion

Multi-sensor fusion can leverage the strengths of various sensors due to their unique characteristics. Specifically, we have integrated LiDAR data into the PMS algorithm. The propagation method was used to fuse the LiDAR depth information and RGB image. Our algorithm significantly improves the accuracy and calculation speed of the PMS algorithm under complex conditions. The method we propose is especially well-suited for complex scenes or objects that are more difficult to estimate depth. In contrast to using a depth camera for depth map calculation, our method is more appropriate for use in large and intricate scenes. Despite its numerous benefits, our method is more reliant on LiDAR data, which may be considered a limitation. Therefore, future research will concentrate on developing depth estimation methods that are less sensitive to LiDAR data.

Data availability statement

The data cannot be made publicly available upon publication because they are not available in a format that is sufficiently accessible or reusable by other researchers. The data that support the findings of this study are available upon reasonable request from the authors.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China under Grant Nos. 62273344 and 61973300.

ORCID IDs

Guangyao Xu  https://orcid.org/0000-0003-1563-8085
En Li  https://orcid.org/0000-0002-4412-2953

References

[1] Zhang Z 2012 Microsoft kinect sensor and its effect IEEE Multimedia 19 4–10
[2] Keselman L, Iselin Woodfill J, Grunnet-Jepsen A and Achintya B 2017 Intel RealSense stereoscopic depth cameras Proc. IEEE Conf. on Computer Vision and Pattern Recognition Workshops pp 1–10
[3] Premebida C, Garrote L, Alireza Asvadi A P R and Nunes U 2016 High-resolution lidar-based depth mapping using bilateral filter 2016 IEEE 19th Int. Conf. on Intelligent Transportation Systems (ITSC) (IEEE) pp 2469–74
[4] Maddern W and Newman P 2016 Real-time probabilistic fusion of sparse 3D lidar and dense stereo 2016 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS) (IEEE) pp 2181–8
[5] Shivakumar S S, Mohta K, Pfommer B, Kumar V and Taylor C J 2019 Real time dense depth estimation by fusing stereo with sparse depth measurements 2019 Int. Conf. on Robotics and Automation (ICRA) (IEEE) pp 6482–8
[6] Wang T-H, Hou-Ning H, Hubert Lin C, Tsai Y-H, Chiu W-C and Sun M 2019 3D lidar and stereo fusion using stereo matching network with conditional cost volume normalization 2019 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS) (IEEE) pp 5895–902
[7] Choe J, Joo K, Imtiaz T and So Kweon I 2021 Volumetric propagation network: stereo-lidar fusion for long-range depth estimation IEEE Robot. Autom. Lett. 6 4672–9
[8] Zhang J, Srinivasan Ramanagopal M, Vasudevan R and Johnson-Roberson M 2020 Listereo: generate dense depth maps from lidar and stereo imagery 2020 IEEE Int. Conf. on Robotics and Automation (ICRA) (IEEE) pp 7829–36
[9] Scharstein D and Szeliski R 2003 High-accuracy stereo depth maps using structured light 2003 IEEE Computer Society
[10] Bleyer M, Rhemann C and Rother C 2011 Patchmatch stereo-stereo matching with slanted support windows Bmvc vol 11 pp 1–11 (available at: www.microsoft.com/en-us/research/wp-content/uploads/2011/01/PatchMatchStereo_BMVCC2011_6MB.pdf)

[11] Zbontar J and LeCun Y et al 2016 Stereo matching by training a convolutional neural network to compare image patches J. Mach. Learn. Res. 17 2287–318

[12] Garg R, Vijay Kumar B, Carneiro G and Reid I 2016 Unsupervised CNN for single view depth estimation: geometry to the rescue European Conf. on Computer Vision (Springer) pp 740–56

[13] Godard C, Mac Aodha O and Brostow G J 2017 Unsupervised monocular depth estimation with left-right consistency Proc. IEEE Conference on Computer Vision and Pattern Recognition pp 270–9

[14] Geiger A, Lenz P, Stiller C and Urtasun R 2013 Vision meets robotics: the kitti dataset Int. J. Robot. Res. 32 1231–7

[15] Barlow H B 1989 Unsupervised learning Neural Comput. 1 295–311

[16] Lowe D G 1987 Three-dimensional object recognition from single two-dimensional images Artif. Intell. 31 355–95

[17] Bartoccioni F, Zablocki Eloi, Pérez P, Cord M and Alahari K 2023 Lidartouch: monocular metric depth estimation with a few-beam lidar Comput. Vis. Image Underst. 227 103601

[18] Godard C, Mac Aodha O, Firman M and Brostow G J 2019 Digging into self-supervised monocular depth estimation Proc. IEEE/CVF Int. Conf. on Computer Vision pp 3828–38

[19] Zhao S, Gong M, Huan F and Dacheng T 2021 Adaptive context-aware multi-modal network for depth completion IEEE Trans. Image Process. 30 5264–76

[20] Guizilini V, Ambrus R, Chen D, Zakhavar S and Gaidon A 2022 Multi-frame self-supervised depth with transformers Proc. IEEE/CVF Conf. on Computer Vision and Pattern Recognition pp 160–70

[21] Chen X, Zhang R, Jiang J, Wang Y, Li G and Li T H 2023 Self-supervised monocular depth estimation: solving the edge-fattening problem Proc. IEEE/CVF Winter Conf. on Applications of Computer Vision pp 5776–86

[22] Park K, Kim S and Sohn K 2019 High-precision depth estimation using uncalibrated lidar and stereo fusion IEEE Trans. Intell. Syst. 21 321–35

[23] Huber D and Kanade T et al 2011 Integrating lidar into stereo for fast and improved disparity computation 2011 Int. Conf. on 3D Imaging, Modeling, Processing, Visualization and Transmission (IEEE) pp 405–12

[24] Sander J, Ester M, Kriegel H-P and Xiaowei X 1998 Density-based clustering in spatial databases: the algorithm GDBSCAN and its applications Data Min. knowl. Discovery 2 169–94

[25] Jason K, Harakeh A and Waslander S L 2018 In defense of classical image processing: Fast depth completion on the cpu 2018 15th Conf. on Computer and Robot Vision (CRV) (IEEE) pp 16–22

[26] Hengjie L, Shugong X and Cao S 2021 SGTBN: generating dense depth maps from single-line lidar IEEE Sens. J. 21 19091–100

[27] Yoon K-J and Kweon I-S 2005 Locally adaptive support-weight approach for visual correspondence search 2005 IEEE Computer Society Conf. on Computer Vision and Pattern Recognition (CVPR'05) vol 2 (IEEE) pp 924–31

[28] Hosni A, Rhemann C, Bleyer M, Rother C and Gelautz M 2012 Fast cost-volume filtering for visual correspondence and beyond IEEE Trans. Pattern Anal. Mach. Intell. 35 504–11

[29] Uhrig J, Schneider N, Schneider L, Franke U, Brox T and Geiger A 2017 Sparsity invariant cnns 2017 International Conference on 3D Vision (3DV) (IEEE) pp 11–20

[30] Wong A and Soatto S 2021 Unsupervised depth completion with calibrated backprojection layers Proc. IEEE/CVF Int. Conf. on Computer Vision pp 12747–56

[31] Tomasi C and Manduchi R 1998 Bilateral filtering for gray and color images Sixth Int. Conf. on Computer Vision (IEEE Cat. No. 98CH36271) (IEEE) pp 839–46

[32] Kato S et al 2018 Autoware on board: enabling autonomous vehicles with embedded systems 2018 ACM/IEEE 9th Int. Conf. on Cyber-Physical Systems (ICCPS) (IEEE) pp 287–96