# Literature Review: Conventional Stereo Correspondence Algorithms

Arjun Ashok Rao
arjunrao@link.cuhk.edu.hk

## Abstract

*The task of establishing left-right correspondence in a stereo pair has been a challenge, and has been attracting greater work in the field every year. Although recent work is saturated with learning-based methods to establish this correspondence, they are often inspired by more conventional stereo correspondence algorithms - both local and global. In this review, we aim to survey these conventional stereo matching algorithms in terms of their matching cost, cost aggregation function, and disparity calculation algorithm. Finally, we discuss the Dynamic Programming and Semi-Global-Matching algorithms by categorizing the algorithms based on the taxonomy in Scharstein [19] to provide a better analysis compared to a top-down explanation.*

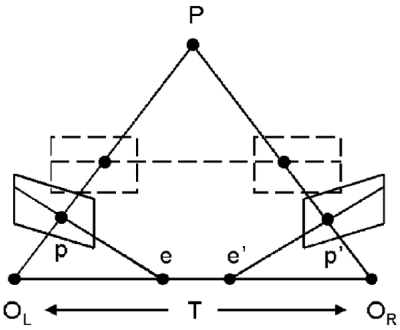



**Figure 1:** $O_L$ and $O_R$ are capturing the image of object P as left and right images $p$ and $p^{'}$ respectively. The dotted planes refer to the rectified planes after performing the stereo rectification algorithm. $pe$ and $p^{'}e^{'}$ are the epipolar lines.

## 1 Introduction

The steps taken to solve the stereo correspondence problem to ultimately generate a disparity map involves formulating a matching cost function, aggregating the cost, computing the disparity, and applying a spatial filter on the produced disparity to further refine the disparity map. These stages are defined in [19], and further expanded on in a structural fashion in [6]. Stereo disparity maps can also be generated through local or global methods. Local methods function by a window- based comparison where the disparity for the target pixel $p_l$ is computed in relation to a chosen window surrounding it. In contrast, global methods aim to minimize a join energy functions which is derived from the entire image. Global approaches are computationally expensive. Stereo matching cost aims to determine the position of a target pixel on the left image $p_l$ on the corresponding right image. A naive approach would be to integrate over every pixel in the reference image to compute the disparity for the target pixel in the left image. However, this is computationally infeasible. Thus, we make use of the epipolar line in the stereo pair for a 1-D search. Stereo rectification is a necessary pre-requisite to establishing stereo correspondence.

### 1.1 Stereo Rectification and Epipolar Constraint

The matching cost function can be defined as the method of determining the parallax values of each pixel between the left and right images. However, it is necessary that both stereo images have epipolar lines that lie along horizontal scan lines. In figure 1, $O_L$ and $O_R$ refer to the left and right cameras attempting to capture an image of object $P$. The corresponding captured images are visible at $p$ and $p^{'}$, and the plane containing this image, is also the plane of projection for the point P. This plane is called the epipolar plane, and the line which connects the epipolar plane to intersect the image planes is called the epipolar line. In figure 1, $pe$ and $p^{'}e^{'}$ constitute the epipolar lines. Using existing algorithms, and by

leveraging the epipolar constraint, it is possible to make the epipolar lines lie along horizontal scanlines. If this is complete, then the stereo pair is said to be rectified. The primary reason behind rectification of a stereo pair if to reduce the search dimensionality. Matching cost computation is reduced to a 1-D search if both stereo pairs are rectified. This also means that each row of pixels in left image $I_L$ corresponds to the same row of pixels in the right image $I_R$.

### 1.2 Previous Reviews on Conventional Stereo Algorithms

This literature review draws on the findings of previous reviews such as [6] and [2]. Hamzah et al. [6] focusses on a review structure based on the general 4-stage taxonomy initially proposed in [19]. Brown et al. [2] instead branch their review into different local and global methods and also focus on conventional algorithm's occlusion detection and handling strategy. Both [2] and [6] discuss the real-time implementations of stereo matching in hardware accelerated systems such as FPGA and GPU.

The rest of this review will be structured as follows - We will branch our review similar to the taxonomy proposed in Scharstein [19] and briefly discuss the mathematical formulae for each component along with discussing one implementation. Finally, we add two sections after the taxonomy - occlusion detection and boundary comprehension. Although recent methods for these challenges include several learning based methods, our review will cover conventional stereo algorithms.

## 2 Matching Costs

Previous reviews on the matching costs include an analysis of matching cost functions done only for stereo pairs with certain radiometric differences ([10]), matching costs reviewed by their local/global property ([2]) and a categorical listing of common matching functions ([6]). We categorize the matching costs based on their computational complexity.

### 2.1 Problem Formulation

Given two stereo rectified images $I_L$ and $I_R$, we consider the pixel to be matched ($p$) and its correspondence with a pixel $p^{'}$ in $I_R$ which lies along the same epipolar line. Since both rectified stereo images lie on horizontal scanlines, $p^{'} = ep(p, d)$ where $d$ is the horizontal disparity which is to be estimated accurately. $ep$ is indicative of the epipolar line, and for rectified stereo pair, $ep(p, d) = [p_x - d, p_y]^T$. Notice that we need to only find the disparity along the horizontal axis, thus making the matching cost computation a 1-D optimization problem. Although matching can be computed on the basis of color and other correlations, we focus on the pixel intensity difference as the primary matching criterion.

### 2.2 Absolute Difference (AD)

The most naive approach to computing the matching cost. AD cost for a pair of pixels can be computed as below:

$$AD(p_x, p_y, d) = |I_L(p_x, p_y) - I_R(p_x - d, p_y)|$$

Computational complexity of this matching is very low due to a single pixel-based matching rather than taking the sum of an entire window/kernel to come up with an optimal cost. Improvements to this include a Truncated Absolute difference proposed in Pham et al. [17]. In this technique, the matching cost is computed as the sum of the absolute difference of the color and user-defined truncation value. The mathematical expression for TAD is [17]:

$$C_d^{Color}(p) = min(\sum_{i=1}^{3} I_i(p) - \hat{I}_i(p - d)|, T_c)$$

Where $T_c$ is the user-specified truncated color value. However, since this metric is not based on Image intensity, we can disregard it for this survey.

### 2.3 Squared Absolute Difference

The pixel-wise Squared difference can be written as:

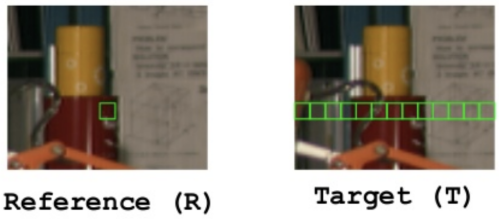
$$SD(p_x, p_y, d) = |I_L(p_x, p_y) - I_r(p_x - d, p_y)|^2$$

**Drawbacks**: Highly sensitive to brightness and noise, highest error rate.

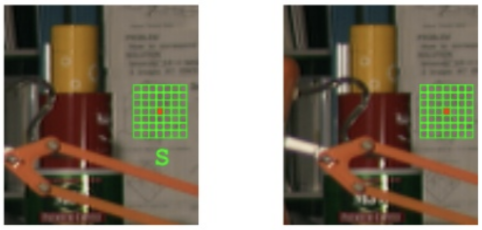
### 2.4 Birchfield and Tomasi

Like AD and SD, Birchfield et al. [1] propose a matching cost algorithm that is sampling insensitive which is achieved by linearly interpolating of the sample. This method is described more clearly in Hirschmüller [8] where the author asserts that Birchfield et al. [1] calculate the sub-pixel-wise cost by shifting in the range of half a pixel in each direction along the epipolar line. Mathematically, this can be easily expressed below. Given two columns along the same horizontal scan-line denoted by $x_l and x_r$, the Birchfield-Tomasi dissimilarity can be computed as:

$$d(x_l, x_r) = min(d_l(s_l, x_r), d_r(x_l, x_r))$$




**(a)** Simplified intuition on Pixel-wise matching methods - a target pixel in the left image is used to compute costs with other pixels in the horizontal scanline of the reference image. Source



**(b)** Simplified intuition on window-wise matching methods - a target pixel surrounded by a matching block is summer or averaged and cost is computed with a similar block in the reference image. Source

where the intermediate function $d(x_l, x_r)$ is given by:

$$d_l(x_l, x_r) = min_{x_r - \frac{1}{2} \leq x \leq x_r + \frac{1}{2}} |I_l(x_l) - \hat{I}_r(x)|$$

$$d_l(x_l, x_r) = min_{x_l - \frac{1}{2} \leq x \leq x_l + \frac{1}{2}} |\hat{I}_l(x) - \hat{I}_r(x_r)|$$

$\hat{I}_l$ and $\hat{I}_r$ are the linear interpolation functions [1]. Similarity is determined by finding the best match between the target pixel in $I_l$ and the interpolated pixel (usually by $\frac{1}{2}$ pixels) in the reference image.

## 2.5 SAD, SSD,NCC

An extension from AD algorithm, this method is a block-matching method which is computed using a filter. (Good Python implementation here). SAD refers to Sum over Absolute differences and SSD refers to Square of Sum Over Absolute Differences. They can be easily formulated below:

$$SAD(p_x, p_y, d) = \sum_{p_x, p_y \epsilon w} |I_L(p_x, p_y) - I_R(p_x - d, p_y)|$$

$$SSD(p_x, p_y, d) = \sum_{p_x, p_y \epsilon w} (I_L(p_x, p_y) - I_R(p_x - d, p_y))^2$$

For both these methods, computation complexity is greater since each window is required to be aggregated in order to match one target pixel. This window aggregation is done inside the nested loop of the image. This also applies to other methods such as NCC, where the formula can be derived easily and is given in [2]. The primary advantage of NCC is that the normalization done in a matching block compensates for the differences in gain and bias [6] A short algorithm for general block based methods can be looked at in Algorithm 1

**Algorithm 1** General Algorithm for Block Based Matching Functions

**Input** $I_L, I_R$ Disparity Offset $O$, Kernel Size $K$
**Output** Computed Disparity Map $D$

INITIALIZE Depth $D \leftarrow 0$
$h \leftarrow height(I_L)$,$w \leftarrow width(I_L)$
**while** $\frac{K}{2} < y < h - \frac{K}{2}$ **do**
  **while** $\frac{K}{2} < x < w - \frac{K}{2}$ **do**
    **while** $-\frac{K}{2} < v < \frac{K}{2}$ **do**
      **while** $-\frac{K}{2} < u < \frac{K}{2}$ **do**
        Difference $\leftarrow I_L[y+v, x+u] - I_R[y+v, x+u-O]$
        Calculate *score* with difference
      **end while**
    **end while**
    **if** score $\leq$ previous_score **then**
      previous_score $\leftarrow$ score
      best $O \leftarrow O$
    **end if**
  **end while**
**end while**

## 2.6 Rank Transform and Census Transform

Unlike the traditional window matching methods, Census transform is not reliant on summing up over all pixels surrounding the target pixel in a matching block. Instead, the method computes a bit-string based on the intensity-ordering difference among each pixel in the block and the target pixel we wish to compute a matching cost for. A description of the Census transform method is given in figure 4

A drawback of census transform is incorrect matching in regions with repetitive structures[6]. This was overcome by adding additional bits to represent differences between the pixel fo interest and its neighboring pixels [6]. However, Census transform still performs well with occlusions and image boundaries.


**Figure 3:** Dichotomy between window size for stereo matching versus the general robustness of the disparity inside the window is shown. Larger window sizes are more robust and smoother. However, a larger window makes an assumption of **constant disparity** in each window which is proven false in regions of depth discontinuities. This leads to blurred object boundaries [8]. Source

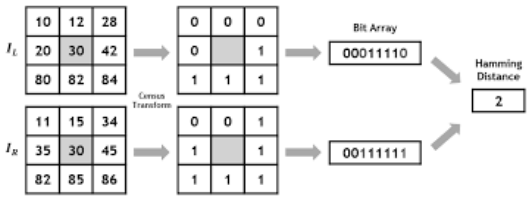



**Figure 4:** Census transform on the two windows in images $I_L, I_R$ are computed by calculating whether each pixel surrounding the centre pixel is larger in intensity than the centre pixel (1 if True, 0 if False). The final bit string is computed by listing the True/False result, and the Hamming distance is calculated as the number of indexes in the list where the bits are differing.

On the other hand, Rank transform is calculated based on an absolute difference between the matrix ranks of $I_L$ and $I_R$. From [6], we can define the rank as:

$$Rank(p_x, p_y) = \sum_{(p_I, p_j)(p_x, p_y)} L(p_I, p_j)$$

$$L(p_I, p_j) = \begin{cases} 0 & I(p_I, p_j) < I(p_x, p_y) \\ 1 & otherwise \end{cases}$$

Here, $p_I, p_j$ are the surrounding pixels in the same image. Once the rank is computed for both left and right images, a simple SAD cost is computed with the ranks instead of the individual pixel intensities.

## 2.7 Feature Based Techniques

In feature based techniques, the correspondence matching cost function is computed with the prerequisite knowledge that certain similar feature points such as edges, shapes, textures, and gradient peaks can be unambiguously matched [6].

### 2.7.1 Scale Invariant Feature Transform (SIFT)

From a stereo matching perspective, SIFT is an algorithm to extract features from images which can later be used to perform matching between the stereo pair. In SIFT, the extracted features are invariant to any kind of image transformation/distortion. Low [14] describes the SIFT algorithm in four stages. In the first stage of the algorithm, the difference-of-Gaussian function is used to identify potential interest points in the image and is iterated over all scales and image locations. Points that are invariant to scale and orientation are chosen. Next, key point localization is performed to determine location and scale. These keypoints are then assigned an orientation(s) based on local image gradient directions. Finally, the local image gradients are measured at the selected scale in the region around each key point. Sharma et al. [20] extend the efficiency of the SIFT algorithm to reduce the computational complexity using a Self-Organizing Map (SOM).

## 2.8 Edge-based and Segment-based cost

Both image segmentation and edge-detection based cost functions have shown promise, but they are insensitive to occlusion and image boundaries. Liu et al. [13] use both segmentation and edge detection to compute matching cost, Çiğla et al. [3] attempts to solve the dense depth-map estimation problem by modeling the scene via non-overlapping planar segments. The segment-based matching cost for this approach refers to planar segments. In much older works such as Park et al. [16], the authors hypothesize that regions of an image with similar color seg-

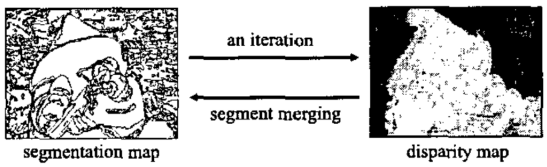



**Figure 5:** Basic functioning of Matching cost of Park et al. [16]. Each local window is selected based on boundaries given by the color segmentation cue. Finally, segments are merged as part of the cost aggregation according to their similar disparities.

ments have similar disparities. Based on this hypothesis, and without a ground truth disparity map, the matching is done using arbitrarily shaped matching regions based on the color segmentation cue [16].

#### 2.8.1 Mutual Information (Global Matching Function)

Mutual information is a global cost function, most popularly used in the Semi-Global Matching algorithm [8]. Egnal [4] offer a good analysis into Mutual information as a stereo correspondence metric. Mathematically, MI relies on the entropy of the probability distribution of the stereo pair. Although this is a global metric, it still requires the computation of a pixel-wise entropy. This entropy is calculated as a joint entropy of two random variables. We can define entropy of a single random variable $\mathcal{X}$ as:

$$H(X) = -\sum_{x \epsilon \mathcal{X}} P(x) \log P(x)$$

Where $P(x)$ is the probability distribution of the random variable, and $\mathcal{X} = [x_1, x_2, ..., x_n]$ . Egnal [4] offer a good intuition of the MI metric from an image processing perspective. The entropy calculated in the equation above measures the randomness of a chosen random variable $\mathcal{X}$. Therefore, entropy is lower in clear, non-occluded regions of the image compared to highly textured and object-border areas. Similarly, the joint entropy $P(\mathcal{X}, \mathcal{Y})$ represents the joint probability function of the LR stereo pair. Two regions with ideal alignment would have a low entropy. However, constant regions in the image will have a low entropy as well. According to [4], this is the main incentive behind maximizing the entropy for a single image. The mathematical formula for MI can be given by:

$$MI(\mathcal{X}, \mathcal{Y}) = H_{\mathcal{X},\mathcal{Y}}(\log(\frac{P(\mathcal{X}, \mathcal{Y})}{P(\mathcal{X})P(\mathcal{Y})}))$$

A pre-requisite for calculating the MI is the probability density functions of the stereo pair. This is done via a histogram with $n = 20$ bins.

# 3 Cost Aggregation

According to Hamzah et al. [6], the main goal of a cost aggregation algorithm is to reduce uncertainties in the matching process. This is done by summing or averaging the computed cost over a support region [19]. Intuitively, getting a good estimate of the matching cost values of a support region surrounding a particular target pixel will give us a better intuition on what disparity we have to assign to the target pixel $(p_x, p_y)$.

Since local methods often establish a pixel-pixel matching cost, cost aggregation is required more for smoothing the local algorithms. However, according to Tombari et al. [22], cost aggregation functions also show promise on global stereo correspondence algorithms. Surveys on common cost aggregation methods have been done in previous works. Tombari et al. [22] offer an in-depth study of cost aggregation methods involving variable windows including windows with adaptive weights and multiple windows along with a comparison based on their accuracy metric. Hamzah et al. [6] offer a categorical analysis mostly focussed on rectangle window-based aggregation. Scharstein [19] offers a very general look at some rectangle and shifting based aggregation methods.

## 3.1 Fixed Window Based Methods

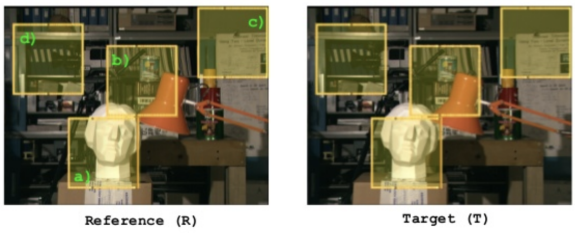
In the Fixed Window method, the support region over which a filter is applied is in 2-D. A simple mathematical representation from Fang et al. [5] is shown below:

$$C_{agg}(x, y, d) = \sum_{\forall (x', y') \epsilon \mathcal{N}(x,y)} C(x^{'}, y^{'}, d)$$

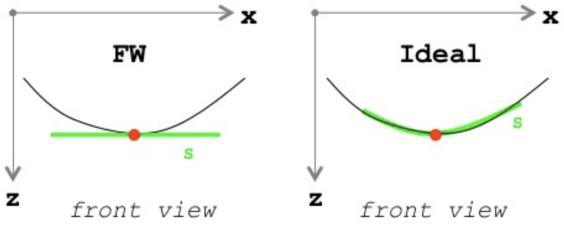
where $\mathcal{N}(x, y)$ refers to all the neighboring pixels of the target pixel $(x, y)$ over the chosen support region.

Although the FW aggregation in computationally inexpensive, it suffers from four main drawbacks:

1. Favorable only for fronto-parallel orientations of a surface of an image (figure 6b)
2. Cannot handle occlusions and depth discontinuities (figure 6a)
3. Cannot handle uniform texture areas (homogeneous areas) (figure 6c)




**(a)** region **a,b,c,d** indicate four regions where using FW aggregation is detrimental to output disparity map accuracy




**(b)** Better look at the fronto-parallel assumption made by Fixed window aggregation for the selected curved surface.

**(c)** Difficulty of fixed window to identify two uniform surfaces (homogeneous surfaces) Source.

**(d)** Fixed window also struggles with repetitive patterns in image due to similar cost for each pattern Source

**Figure 6:** The FW aggregation has four major drawbacks shown in subfigure **5a** - region **a** is due to FW's assumption that the selection region is fronto-parallel which is often incorrect for curved surfaces, **b** is the fixed-window drawback due to ignoring depth discontinuity (occlusion) between lamp and bookshelf, **c** is the region of uniform texture (refer figure c), and **d** is the repetitive pattern drawback (clear in figure d)

4. Cannot handle repetitive patterns in images (figure 6d)

The fixed window strategy can be extended to form a shifted-window aggregation where the pixel is moved around the support window and the position with the least matching cost is chosen. The method does not change computational complexity since the shifted window just translates to a new fixed window with a different pixel position.

## 3.2 Multiple windows (MW), Adaptive Windows (AW)

First proposed in Innocent et al. [11], the MW approach a number of "candidate" windows are chosen along with the original window containing the target pixel. Each candidate window is the same size, but different positions relative to the point of interest [11]. For a total window number of $n = 9$, four windows can be considered at a time, and all permutations of cost are aggregated. 9 window orientation can be seen in figure 7. Correlation is done for all nine windows, and best cost is chosen.

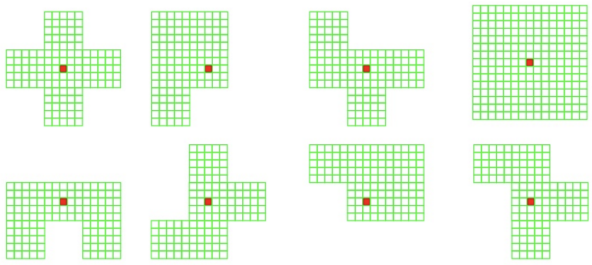


**Figure 7:** Multiple support windows (MW) aggregation with total number of window combinations. The support windows can be added to corners, edges, etc. This method marginally improves performance on depth discontinuities. Each window in the figure is $(5X5)$ pixels

To build on MWs drawbacks on image boundaries, Adaptive Weights(AW) based aggregation selects the support region based on the local structure of the reference image [6]. This reconstruction of local structure helps preserve corners and edges. It achieves this reconstruction by using similarity and proximity based "weights", and the similarity is calculated by taking a Euclidean distance between the pixel of interest and a pixel in the candidate window. [5].

Further fundamental additions to AW has been made such as Adaptive Support weights (ASW) [5], segmentation based aggregation, Bilateral Filtering [21]. For the sake of conciseness, we do not expand on these newer methods.

# 4 Computation of Disparity

After the computation of the matching cost and aggregating of the cost (most computationally expensive stage), computing the disparity is a trivial task. In local methods, the disparity is selected through a Winner Takes All (WTA) optimization. Mathematically, this can be simply expressed as: [6]

$$disparity_p = arg \min_{d \epsilon D} C_{agg}(p, disparity)$$

Where $p$ is the target pixel, $D$ is the allowed range of disparities (Middlebury dataset has different allowed ranges of disparities for different test images), and $C_{agg}$ is the aggregated cost function computed using any of the methods above. Any error due to occlusion is filtered and regularized in the disparity refinement stage. However, the WTA optimization does not necessarily result in a 1-1 pairing developed between disparities in $I_L$ and $I_R$. This 1-1 pairing rule is violated more in regions of common texture such as figure 6c. Therefore, we turn to the global computation of disparity.

In Global Methods, the cost aggregation step is skipped. Instead of aggregating the cost over a Fixed/ moving window, the algorithm's objective to to minis a certain energy function. Rather than calculate a pixel-wise disparity $d$, a generalized disparity function $E(d)$ is calculated as:

$$E(L) = E_{matching}(L(I) = d) + E_{smooth}(L(I), L(j))$$

Here, $E_{matching}$ is the matching cost energy function. For simplicity, $E_{matching} = C_{BT}$ or $E_{matching} = C_{AD}$ where $C_{BT}$ is Birchfield-Tomasi cost function [1] and $C_{AD}$ is an Absolute difference cost function. $L$ is the disparity number assigned, and $L(i) \epsilon [0, d_{max}]$. The smoothness term takes pixel value which is in close proximity to the target pixels. Instead of explaining the different energy minimization functions, we highlight the entire workflow of stereo matching by explaining a few popular stereo matching algorithms below.

# 5 Stereo matching algorithms

In this section, we look at two popular stereo matching algorithms. First, we look at Dynamic Programming methods used in Okutomi et al. [15] and highlight the drawbacks of this approach. Finally, we take a look at Semi-Global ([8]) Matching which uses both local and global techniques to generate accurate stereo correspondence.

Although there are several local algorithms for stereo matching, a majority of these follow a highly similar algorithm, and do not achieve results comparable to global methods. For example, previously mentioned correlation based methods such as Hirschmüller et al. [9] compute matching cost, aggregate the matching cost using a square window of 9X9 pixels, and use a WTA optimization with minimum computed aggregation cost. After computing the disparity, local algorithms go through a series of post processing steps. These are:

1. Occlusion Handling: A left-right consistency check is performed whereby disparities present in one image and not present in the other are classified as occlusions.

2. Disparity clipping: The disparity segments smaller than 160 pixels are invalidated [7] and filled with background disparity values to ensure smoothness.

## 5.1 Dynamic Programming

In Dynamic Programming first proposed in [15], each horizontal scanline is taken as an independent entity. There is no additional smoothness of occlusion constraint. At the heart of DP approach is the condensing of the larger correspondence problem into multiple sub-problems, and the cost of the optimal path is the sum of the costs of the partial paths obtained recursively [2]. This cost finding can be written is a trivial fashion as:

$$d(x_i, y_i) = \min C(x_i, y_i, d)$$

The Dynamic programming proposed in Okutomi et al. [15] can also be thought of as a shortest path finder between two corners for an individual scanline. This is illustrated in figure 10b. Two ways by which we can reduce the complexity of this algorithm is by reducing the range of disparity values that can be assigned to a pixel, and by "forgoing optimality" [2]. This can be exercised by performing a greedy search instead of a continuous iterative approach. The two primary drawbacks of Dynamic Programming are:

1. Streaky artefacts on disparity maps: As a result of matching each scanline individually without smoothness, the resultant disparity map of Okutomi et al. [15] shows horizontal streaks (figure 8).

2. 1-1 matching broken in case of orientation changes in stereo image pair: An essential constraint in the shortest-path dynamic programming approach is to produce a 1-1 matching

**Figure 8:** Streaky artefacts produced in Dynamic Programming based disparities. Source

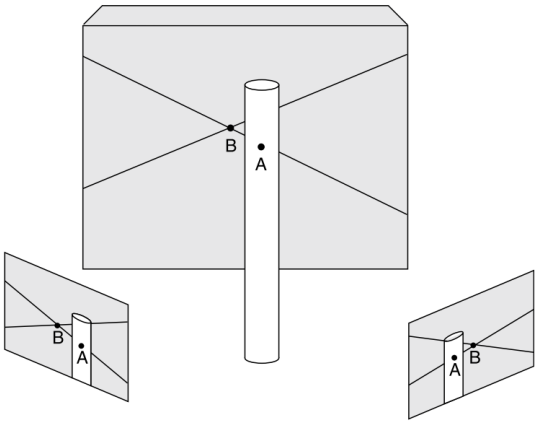
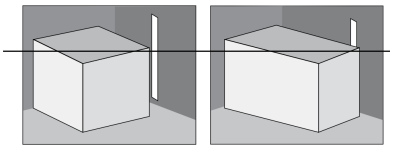


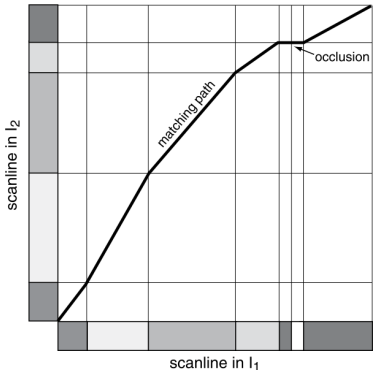

**Figure 9:** The double nail illusion illustrated in [18]. Here, the point A and point B appear with different orders in each pair. In this case, the dynamic programming shortest path will loop backward/ decrease which violates the 1-1 mapping.

graph. In figure: 9, each image is oriented differently with respect to the background. Radke [18] defines this as the double nail illusion.

3. Monotonicity Constraint: DP assumes an ordering constraint - that a pixel on the reference image $I_R$ can only appear at the same index, or a greater index than the pixel on the left image $I_L$. This assumption that both scanlines are oriented in the same order is incorrect in many cases. (9)



**(a)** Horizontal matching with occlusion for DP




**(b)** Shortest path approach taken by DP

**Figure 10:** Subfigure **a** shows a stereo pair with multiple planar surfaces which is traversed by the horizontal scanline. Not all regions present in $I_L$ is present in $I_R$. Subfigure **b** shows the shortest path distance done by Okutomi et al. [15]. The Dynamic programming approach ensures the 1-1 matching constraint for each pixel along the horizontal scanline. The occlusion is also handled well (horizontal path in subfigure **b**) Source.

## 5.2 Semi Global Matching (SGM) [8]

SGM algorithm, initially proposed in Hirschmüller [8] proposes two interesting and novel contributions.

### 5.2.1 Hierarchical MI

From previous sections, MI for two random variables can be calculates as:

$$MI(\mathcal{X},\mathcal{Y}) = E_{\mathcal{X},\mathcal{Y}}(\log(\frac{P(\mathcal{X},\mathcal{Y})}{P(\mathcal{X})P(\mathcal{Y})}))$$

The joint entropy of two stereo images $I_L$ and $I_R$ can be calculated as:

$$H_{I_L,I_R} = -\int_0^1\int_0^1 P_{I_L,I_R}(i_L,i_R)\log P_{I_L,I_R}(i_L,i_R)di_Ldi_R \tag{1}$$

and the joint MI of two images with individual MI $MI_L$ and $MI_R$ is:

$$MI_{I_L,I_R} = H_{I_L} + H_{I_R} - H_{I_L,I_R}$$

According to Kim et al. [12], the joint MI value measures the similarity between two images and can handle complex intensity differences (since it operates solely on the probability distribution of the entropy). The authors are able to transform the MI problem to a global energy minimization problem. They achieve this by using a Taylor approximation to convert the MI problem to a sum of pixels. Graph Cuts is then used to maximize this.

Kim et al. [12] also suggest an iterative ($i = 3$ usually) approach to refining the disparity map where matching cost for each pixel is computed iteratively, and each disparity map is computed given the previous iteration's disparity map. Hirschmüller [8] assert that this is runtime expensive and propose a **Hierarchical method**. This method recursively uses an upsampled disparity image calculated at half-resolution. The initial half-resolution, upsampled disparity image is passed through the same iterative process of Kim et al. [12]. This gives a slightly better runtime.

### 5.2.2 Recursive Cost Aggregation

The energy function in SGM adds two parameters which penalize the disparity calculation.

$$E(D) = \sum_{p}(C(p, D_p) + \sum_{q \epsilon N_p} P_1 T[|D_p - D_q| = 1] + \sum_{q \epsilon N_p} P_2 T[|D_p - D_q| > 1|) \tag{2}$$

The intuition behind penalty $P_1$ is to add a cost in case the exact disparity matching is not achieved (small disparity difference) and $P_2$ is applied in case of larger disparity differences ($P_1 \leq P_2$). The larger penalty is to ensure sharp depth discontinuities are preserved. The smaller penalty is meant for curved surfaces where the depth change is small and uniform. [8]

Finally, SGM aggregates cost from 8 different directions recursively. Mathematically, the recursive cost aggregation can be expressed by: [8].

$$\begin{aligned} L_r^{'}(p, d) = C(p, d) + \min(L_r^{'}(p - r, d)), L_r^{'}(p - r, d - 1) \\ + P_1, L_r^{'}(p - r, d + 1) \\ + P_1, \min(L_r^{'}(p - r, i) + P_2 \end{aligned} \tag{3}$$

Where $P_1, P_2$ are the penalties described, r is the direction of traversal we are performing, $i$ is the disparity value of the current iteration ($i \epsilon [0, d_{max}]$), $L_r^{'}(p, d)$ is the cost associated with the particular chosen path.

An example of this is shown in figure 11.

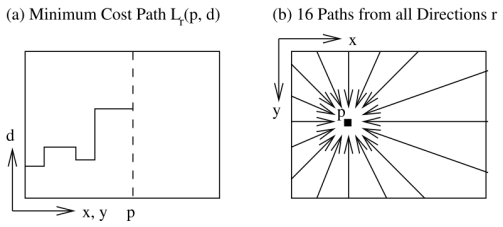



**Figure 11:** In SGM, costs are aggregated from 8 different directions recursively. The aggregated cost is obtained by summing all the costs of the 1-D minimum cost paths that end with the target pixel. Source: [8].

# 6 Conclusion

Stereo matching undoubtedly is a very exciting problem to solve. Although conventional matching algorithms are quite dated, they still serve as a strong foundation for newer research in stereo and computer vision. The rise of dual-camera mobile phones has caused a large increase in stereo-image datasets. Stereo correspondence is a fundamental problem of stereo vision, and is the preliminary building block to other exciting applications such as stereo object detection, stereo super resolution, etc. Thus, solving this problem can help computer vision tremendously.

# References


[1] Stan Birchfield and Carlo Tomasi. "Depth discontinuities by pixel-to-pixel stereo". In: *International Journal of Computer Vision* 35.3 (1999), pp. 269–293. ISSN: 09205691. DOI: 10.1023/A:1008160311296.

[2] Myron Z. Brown, Darius Burschka, and Gregory D. Hager. "Advances in computational stereo". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 25.8 (2003), pp. 993–1008. ISSN: 01628828. DOI: 10.1109/TPAMI.2003.1217603.

[3] Cevahir Çiğla, Xenophon Zabulis, and A. Aydin Alatan. "Region-based dense depth extraction from multi-view video". In: *Proceedings - International Conference on Image Processing, ICIP* 5 (2007), pp. 213–216. ISSN: 15224880. DOI: 10.1109/ICIP.2007.4379803.

[4] Geoffrey Egnal. "Mutual Information as a Stereo Correspondence Measure". In: *Technical Reports, Departement of Computer & Information Science, University of Pennsylvania* MS-CIS-00-20 (2000). URL: http://repository.upenn.edu/cis%7B%5C_%7Dreports/113/.

[5] Jianbin Fang, Ana Lucia Varbanescu, Jie Shen, Henk Sips, Gorkem Saygili, and Laurens Van Der Maaten. "Accelerating cost aggregation for real-time stereo matching". In: *Proceedings of the International Conference on Parallel and Distributed Systems - ICPADS* (2012), pp. 472–481. ISSN: 15219097. DOI: 10.1109/ICPADS.2012.71.

[6] Rostam Affendi Hamzah and Haidi Ibrahim. "Literature survey on stereo vision disparity map algorithms". In: *Journal of Sensors* 2016 (2016). ISSN: 16877268. DOI: 10.1155/2016/8742920.

[7] Heiko Hirschm and Daniel Scharstein. "Evaluation of Cost Functions for Stereo Matching". In: (2007).

[8] Heiko Hirschmüller. "Stereo processing by semiglobal matching and mutual information". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30.2 (2008), pp. 328–341. ISSN: 01628828. DOI: 10.1109/TPAMI.2007.1166.

[9] Heiko Hirschmüller, Peter R. Innocent, and Jon Garibaldi. "Real-time correlation-based stereo vision with reduced border errors". In: *International Journal of Computer Vision* 47.1-3 (2002), pp. 229–246. ISSN: 09205691. DOI: 10.1023/A:1014554110407.

[10] Heiko Hirschmüller and Daniel Scharstein. "Evaluation of cost functions for stereo matching". In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (2007). ISSN: 10636919. DOI: 10.1109/CVPR.2007.383248.

[11] P. R. Innocent, H. Hirschmuller, and J. M. Garibaldi. "Real-time correllation - based stereo vision with reduced error borders". In: *International Journal of Computer Vision* 47 (2002), pp. 229–246.

[12] Junhwan Kim, Vladimir Kolmogorov, and Ramin Zabih. "Visual correspondence using energy minimization and mutual information". In: *Proceedings of the IEEE International Conference on Computer Vision* 2 (2003), pp. 1033–1040. DOI: 10.1109/iccv.2003.1238463.

[13] Jing Liu, Xinzhu Sang, Changxin Jia, Nan Guo, Yangdong Liu, and Guozhong Shi. "Efficient stereo matching algorithm with edge-detecting". In: *Optoelectronic Imaging and Multimedia Technology III*. Vol. 9273. International Society for Optics and Photonics. 2014, p. 927335.

[14] David G Low. "Distinctive image features from scale-invariant keypoints". In: *International Journal of Computer Vision* (2004), pp. 91–110. URL: https://www.cs.ubc.ca/%7B~%7Dlowe/papers/ijcv04.pdf.

[15] Masatoshi Okutomi and Takeo Kanade. "A locally adaptive window for signal matching". In: (1990), pp. 190–199. DOI: 10.1109/iccv.1990.139519.

[16] Sang Yoon Park, Sang Hwa Lee, and Nam Ik Cho. "Segmentation based disparity estimation using color and depth information". In: *Proceedings - International Conference on Image Processing, ICIP* 2 (2004), pp. 3275–3278. ISSN: 15224880. DOI: 10.1109/icip.2004.1421813.

[17] Cuong Cao Pham and Jae Wook Jeon. "Domain Transformation-Based Efficient Cost Aggregation for Local Stereo Matching". In: 23.7 (2013), pp. 1119–1130.

[18] Richard J Radke. *Computer vision for visual effects*. Cambridge University Press, 2013.

[19] Daniel Scharstein. "A Taxonomy and Evaluation of Dense Two-Frame Stereo". In: 47.1 (2002), pp. 7–42.

[20] Kajal Sharma, Kwang-young Jeong, and Sung-Gaun Kim. "Vision based autonomous vehicle navigation with self-organizing map feature matching technique". In: *2011 11th International Conference on Control, Automation and Systems*. IEEE. 2011, pp. 946–949.

[21] C. Tomasi and R. Manduchi. "Bilateral filtering for gray and color images". In: *Proceedings of the IEEE International Conference on Computer Vision* (1998), pp. 839–846. DOI: 10.1109/iccv.1998.710815.

[22] Federico Tombari, Stefano Mattoccia, Luigi Di Stefano, and Elisa Addimanda. "Classification and evaluation of cost aggregation methods for stereo correspondence". In: *26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR* (2008). DOI: 10.1109/CVPR.2008.4587677.