Constrained Optimization for Plane-Based Stereo

Shahnawaz Ahmed, Miles Hansard, Andrea Cavallaro

***Abstract*—Depth and surface normal estimation are crucial components in understanding 3D scene geometry from calibrated stereo images. In this paper, we propose visibility and disparity magnitude constraints for slanted patches in the scene. These constraints can be used to associate geometrically feasible planes with each point in the disparity space. The new constraints are validated in the PatchMatch Stereo framework. We use these new constraints not only for initialization, but also in the local plane refinement step of this iterative algorithm. The proposed constraints increase the probability of estimating correct plane parameters and lead to an improved 3D reconstruction of the scene. Furthermore, the proposed constrained initialization reduces the number of iterations to converge to the optimum plane parameters. In addition, as most stereo image pairs are not perfectly rectified, we modify the view propagation process by assigning the plane parameters to the neighbours of the candidate pixel. To update the plane parameters in the plane refinement step, we use a gradient free non-linear optimiser. The benefits of the new initialization, propagation and refinement schemes are demonstrated.**

***Index Terms*—Binocular vision, stereo matching, PatchMatch, sub-pixel disparity, surface normal, BOBYQA**

I. Introduction

ACCURATE depth estimation from stereo images is important in many applications, such as augmented reality [1], terrain estimation [2], mapping [3], navigation [4], scene segmentation [5], object recognition [6] and 3D reconstruction [7]. In addition to depth, surface orientation information is important for understanding the scene geometry. The PatchMatch Stereo (PMS) framework [8] can simultaneously estimate the depth and the surface normal at every pixel in an image. Algorithms [9] [10] [11] using the PMS framework reconstruct the surface of the scene in the disparity space [12] defined by the pixel coordinates and the possible disparities. The reconstruction is based on associating a slanted plane with each candidate match, in contrast with methods that evaluate the complete disparity space image either explicitly or by searching over the range of disparities at each pixel.

The key ideas of the PMS framework are that neighbouring pixels have coherent matches, and large numbers of random samples will yield some good initial estimates of the plane parameters. The framework randomly assigns the plane parameters to each pixel of both images and later uses two kinds of propagation scheme; *spatial* and *view* to propagate the correct plane parameters within and across images, followed by an optimization scheme called *plane refinement*. Spatial propagation propagates the good plane parameters among spatial neighbours whereas view propagation propagates the plane parameters within and across views. Plane parameters are also locally optimized at individual pixels in the plane refinement step. The PMS framework produces two separate disparity maps for the stereo pair. Note that, the framework also assumes that the image pairs are rectified, which reduces the stereo correspondence problem to a 1-D search problem, where matching points lie along the horizontal scan-line of the rectified images.

The basic PMS framework has four limitations. First, its initialization process does not guarantee a geometrically feasible plane at each pixel, as it randomly selects the plane parameters for each pixel in both reference and search images. Second, the plane refinement process uses a variant of the Luus-Jaakola optimization [13] to minimize the cost function, which is inefficient, and not guaranteed to find a local minimum of the given cost function. Third, the framework assumes that the stereo images are perfectly rectified which is not the case for typical stereo pairs. Last, it may generate false matches in low textured areas [14].

In this paper, we present a constrained initialization scheme that works with any algorithm that can be cast in the PMS framework. We introduce two new constraints to restrict the initialization scheme by generating only geometrically feasible planes such that the disparity of every pixel inside a patch must lie between the maximum and minimum allowed disparity. The proposed constraints are also imposed during the optimization process. For the plane refinement problem, the usual PMS cost function cannot be minimised by standard gradient descent methods due to the presence of discontinuous thresholds in the pixel dissimilarity function [8]. We avoid this problem by using the "Bound Optimisation BY Quadratic Approximation" (BOBYQA) algorithm [15]. BOBYQA is a gradient free non-linear trust region based constrained optimiser [16] which uses our geometric constraints to ensure that the plane parameters remain feasible. Moreover, we relax the view propagation by assigning the plane parameters to the immediate 4-neighbours (left, right, upper and lower) of the candidate pixel to tackle the imperfectly rectified image pairs. This strategy is also useful as most plane parameters are incorrect in the earlier propagations. To make the cost function more robust to false matches in occluded regions, we change the support weight function so that it considers both search and reference image weights. We finally update the pixel dissimilarity function from the truncated sum of absolute colour difference (TSAD) to the truncated sum of square difference (TSSD), which is more compatible with BOBYQA.

The rest of the paper is organised as follows. Section II discusses the related work. The general framework is proposed in Section III. We introduce our constrained initialization in

S. Ahmed, M. Hansard and A. Cavallaro are with the Centre for Intelligent Sensing, Queen Mary University of London, Mile End Road, UK, E1 4NS, emails: shahnawaz.ahmed@qmul.ac.uk, miles.hansard@qmul.ac.uk, a.cavallaro@qmul.ac.uk.

Section IV and the constrained optimization in Section V. Section VI provides information on the experimental set-up, parameters used and the results. Finally, conclusions are drawn in Section VII.

II. Related Work

Stereo matching algorithms can be broadly classified as local or global methods. *Local methods* are driven by the compatibility of individual matches. *Global methods* make explicit smoothness assumptions and minimize a global cost function. The extra smoothness assumption helps global methods to produce a more accurate disparity map compared to the local ones, but they often require longer execution time [17].

Most local methods involve four stages for computing the disparity map: matching cost computation, cost aggregation, disparity computation and disparity refinement [12]. A cost function is used to measure the matching cost, *i.e.,* the apparent dissimilarity across views. Common pixel-based matching costs are based on the absolute or squared difference of colour intensities. Since the colour intensities may be misleading, image gradients are often combined with the cost function [18]. The disparity is selected by comparing the aggregated matching cost at different disparity values and using the winner-takes-all (WTA) strategy. Finally, some post-processing is performed to fill in the occluded pixels and mismatches via a left/right consistency check [12] in the disparity refinement step. Filling the occluded regions often generates artefacts in the disparity map. To solve this problem, filters (e.g. weighted median) are applied on the disparity map for smoothing the artefacts [19], [20]. A comparison of the current stereo methods can be found on the Middlebury stereo webpage [21].

Most local methods assume implicitly that all the pixels inside the support window have constant disparity (that of the center pixel) [22], [23], [24]. This assumption makes the patch size an important parameter that greatly influences disparity accuracy and computational time. Even though small patches preserve the disparity discontinuities and make the algorithm computationally efficient, other problems may arise, such as the patch not being able to capture enough texture variation to solve matching ambiguities. On the other hand, large patches produce a smoother disparity map, but with the inherent drawback of badly preserved object borders. In the case of depth discontinuities, the smoothness assumption is broken as the patch captures both foreground and background regions, which leads to the 'edge fattening' problem [7]. To overcome this problem, the adaptive support weight [25] was proposed, where the influence of the pixels inside the patch depends on the colour and spatial similarity with the center pixel of the patch. This strategy reduces the edge fattening problem, while retaining the benefits of a large patch size.

The smoothness assumption in local methods does not hold in scenarios where the support window contains pixels that lie on a different surface than that of the centre pixel, or where the support window captures a non-fronto-parallel surface (surface not parallel to the image plane), such as a slanted or curved surface. The adaptive support weight provides the solution for the first problem and has already been implemented in [26] and [20]. For the latter problem, Gallup et al. [27] presented the idea of using the slanted planes in disparity space instead of fronto-parallel ones and proposed a real time multi-view global method generating a sparse disparity map. Bleyer et al. [8] combined the slanted support and PatchMatch [28] in PMS. PMS is an iterative algorithm that relies on random sampling and propagation of good plane parameter estimates.

PMS reconstructs the surface of the scene in the disparity space [12] by associating a slanted plane with each pixel of an image. While assigning the plane parameters randomly to each pixel of an image, there is a high probability that at least one of the pixels will have approximately correct plane parameters. PMS computes an individual plane in the disparity space at each pixel onto which the support region is projected. PMS only requires a single plausible estimation of the plane parameters, which is then propagated to the neighbouring pixels and across images.

Li and Zucker [29] presented a general surface model which fits curved surfaces at each pixel in the scene space. The scene space was used because the disparity space has numerical sensitivity problems with higher order derivatives. The planar model can be regarded as a special case of this general model which works in disparity space. Planar models deal with only first-order derivatives, so slanted planar surfaces are stable in the disparity space.

Many local methods derive the sub-pixel information in the post processing step by fitting a parabola to the cost function, whereas PMS computes the sub-pixel information directly. Besse et al. [10] unified the Particle Belief Propagation (PBP) and PatchMatch and formulated a global stereo algorithm, PatchMatch Belief Propagation (PMBP). PMBP uses the same cost function of PMS with an additional smoothness function that measures the deviation between two local planes. Heise et al. [9] presented an explicit variational smoothness model for PMS using quadratic relaxation [30] and the same smoothness term used in PMBP. State-of-the-art stereo methods using slanted planes are summarised in Table I.

III. General framework

Here we formulate a general setting for PMS, as well as for our contributions in Section IV and V. We begin by defining the relationships between planes in disparity space. We also present modifications to the cost function and the propagation procedure in Section III-B and III-C2.

A. Point-normal plane representation in disparity space

To find corresponding pixels, PMS starts with a rectified colour stereo pair, comprising I and I', where I is the reference image and I' is the search image (which will be exchanged during the course of the algorithm). Let $\mathcal{S}$ be the visible surface. Let $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ be corresponding pixels projected from a scene point $\mathbf{P} \in \mathcal{S}$ to the image plane I and I', respectively (Fig. 1). Let $\tilde{\mathbf{n}}$ be the outward unit surface normal of the plane $\tilde{\mathbf{f}}$ in the scene space containing $\mathbf{P}$. We can find the depth of $\mathbf{P}$ provided the plane $\tilde{\mathbf{f}}$ is known, *i.e.,* both $\mathbf{P}$ and $\tilde{\mathbf{n}}$ are known. As disparity is inversely proportional

TABLE I
COMPARISON OF STEREO MATCHING ALGORITHMS USING SLANTED PLANES

Type	Algorithm	Cost function		Adaptive support weight		Smoothness	No. of iterations
		Intensity	Gradient	Reference image weight	Search image weight		
Global	Gallup et al. [27]	✓				✓	–
	PMBP [10]	✓	✓	✓		✓	3
	PM-Huber [9]	✓	✓	✓		✓	3
Local	PMS [8]	✓	✓	✓			3
	IPMS (proposed)	✓	✓	✓	✓		2

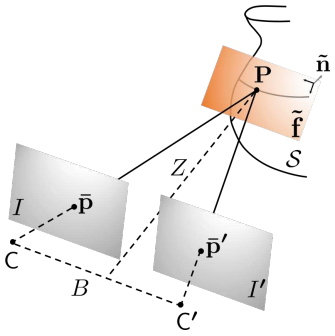

Fig. 1. Pinhole camera model. Image points $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ are the projections of a scene point $\mathbf{P} \in \mathcal{S}$ on the reference image I and the search image I' from two different views obtained by the left camera C and the right camera C' respectively, where $\mathcal{S}$ is the visible surface. The baseline distance between C and C' is B. The scene point $\mathbf{P}$ is at a distance Z from B. The plane $\tilde{\mathbf{f}}$ at $\mathbf{P}$ has a unit surface normal $\tilde{\mathbf{n}}$ in the outwards direction. Our objective is to find Z and $\tilde{\mathbf{n}}$ from the plane $\tilde{\mathbf{f}}$.

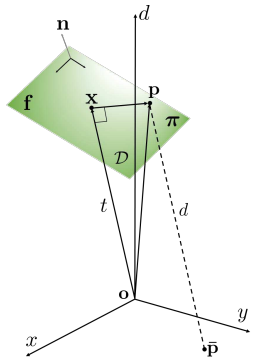

Fig. 2. Representation of a plane in the disparity space. Point $\mathbf{p}$ is the corresponding point in the disparity space $\mathcal{D}$ of an image point $\bar{\mathbf{p}}$ with disparity d. The plane $\mathbf{f}$ at $\mathbf{p}$ has a unit surface normal $\mathbf{n}$ in the outward direction. The plane $\boldsymbol{\pi} = (\mathbf{n}, -t)$ is the point-normal representation on $\mathbf{f}$, where t is the projection of $\mathbf{op}$ on $\mathbf{n}$. The value of t can be positive or negative depending on the orientation of $\mathbf{f}$.

to depth, disparity can also be found by the analogous plane representation of $\tilde{\mathbf{f}}$ in the disparity space. Note that the real world planes are different from the planes in the disparity space as they are related by a projective transformation [29].

Let $\bar{\mathbf{p}} = (x, y)^\mathsf{T} \in I$ and $\bar{\mathbf{p}}' = (x', y)^\mathsf{T} \in I'$. As I and I' are rectified and $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ are matching pixels, $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ have the same y coordinate and the same disparity magnitude d. Then $\mathbf{p} = (x, y, d)^\mathsf{T} \in \mathcal{D}$ and $\mathbf{p}' = (x', y, d)^\mathsf{T} \in \mathcal{D}'$ are two different projective transformations of $\mathbf{P}$ [9], where $\mathcal{D}$ and $\mathcal{D}'$ denote the disparity spaces generated by the reference and search image pixels, respectively. We have used different notations for the disparity spaces to highlight the projective transformations of $\mathbf{P}$ generating from two different views. As $\bar{\mathbf{p}} \in I$, $x' = x - d$, otherwise $x' = x + d$. The relation between $\mathbf{p}$ and $\mathbf{p}'$ in the disparity space is:

$$\mathbf{p}' = \mathbf{M}\mathbf{p}, \quad \text{where} \quad \mathbf{M} = \begin{pmatrix} 1 & 0 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} . \tag{1}$$

Disparity spaces $\mathcal{D}$ and $\mathcal{D}'$ will differ due to the visibility effects. *i.e.,* eq. 1 is not true for all points in practice. Using the projective transformations of $\mathbf{P}$, the scene surface normals can be found via the surface normals in the disparity space.

A plane $\mathbf{f}$ in the disparity space is defined by a point $\mathbf{p}$ and a surface normal $\mathbf{n}$ at $\mathbf{p}$ (Fig. 2). When $\mathbf{p}$ and $\mathbf{n}$ are known, $\mathbf{f}$ can be represented by three plane parameters $\mathbf{f} := (a, b, c)^\mathsf{T}$. The disparity d of $\bar{\mathbf{p}}$ with respect to the plane $\mathbf{f}$ is given by:

$$d = ax + by + c \ . \tag{2}$$

Therefore, disparity is *over parametrized* by the plane parameters of $\mathbf{f}$. If $(a, b, c)^\mathsf{T}$ is known, we can find the disparity δ of any pixel $(\xi, \eta)^\mathsf{T}$ with respect to the plane $\mathbf{f}$. The disparity δ is given by

$$\delta = a\xi + b\eta + c \ .$$

To find boundary constraints on the disparity and the surface normal, our proposed method, *Initialised PatchMatch Stereo* (IPMS), works with the point-normal parametrisation of planes (Fig. 2); unlike PMS which directly uses the plane parameters.

Let $\mathbf{n} = (u, v, w)^\mathsf{T}$ be the unit surface normal at $\mathbf{p}$. In homogeneous coordinates, the plane passing though $\mathbf{p}$ with normal $\mathbf{n}$ can be defined as

$$\boldsymbol{\pi} := (\mathbf{n}, -t) \cdot (\mathbf{x}, 1) = 0, \qquad \text{where } t = \mathbf{n} \cdot \mathbf{p} \ .$$

We use the notation $\mathbf{f}$ and $\boldsymbol{\pi}$ when the plane is parametrised by plane parameters and point-normal, respectively. The relation between the plane parameters $\mathbf{f} := (a, b, c)^\mathsf{T}$ and the unit normal $\mathbf{n} = (u, v, w)^\mathsf{T}$ at $\mathbf{p}$ is:

$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} = \frac{1}{w} \begin{pmatrix} -u \\ -v \\ t \end{pmatrix} , \qquad \mathbf{n} = \frac{1}{\sqrt{1 + a^2 + b^2}} \begin{pmatrix} -a \\ -b \\ 1 \end{pmatrix} .$$

B. Cost function

The performance of a stereo matching algorithm depends on matching cost for measuring the similarity of an image region across views [31]. The intensities of a matching region can differ because of certain radiometric changes and/or noise. Therefore, a matching cost function has to be robust to such variations. We use a pixel-based matching cost function along with adaptive support weight [25].

Let $\mathbf{p}$ be a point in the disparity space which corresponds to an image point $\bar{\mathbf{p}}$. Let Π be the set of all candidate planes passing through $\mathbf{p}$, we want to find a plane $\boldsymbol{\pi}$ that minimises the aggregated matching cost:

$$\boldsymbol{\pi} = \underset{\boldsymbol{\omega}\in\Pi}{\arg\min}\ \mathrm{cost}\big(\bar{\mathbf{p}}, \boldsymbol{\omega}\big)\ .$$

The aggregated cost of $\bar{\mathbf{p}}$ according to $\boldsymbol{\pi}$ is computed as

$$\mathrm{cost}(\bar{\mathbf{p}}, \boldsymbol{\pi}) = \frac{\sum\limits_{\bar{\mathbf{q}}\in\mathcal{W}(\bar{\mathbf{p}})}\ \sum\limits_{\bar{\mathbf{q}}'\in\mathcal{W}(\bar{\mathbf{p}}')} A\big(\bar{\mathbf{p}}, \bar{\mathbf{q}}\big)\cdot A\big(\bar{\mathbf{p}}', \bar{\mathbf{q}}'\big)\cdot E\big(\bar{\mathbf{q}}, \bar{\mathbf{q}}'\big)}{\sum\limits_{\bar{\mathbf{q}}\in\mathcal{W}(\bar{\mathbf{p}})}\ \sum\limits_{\bar{\mathbf{q}}'\in\mathcal{W}(\bar{\mathbf{p}}')} A\big(\bar{\mathbf{p}}, \bar{\mathbf{q}}\big)\cdot A\big(\bar{\mathbf{p}}', \bar{\mathbf{q}}'\big)}\ , \tag{3}$$

where $\mathcal{W}(\bar{\mathbf{p}})$ denotes a square patch centred at $\bar{\mathbf{p}}$, $\bar{\mathbf{p}}'$ denotes the matching pixel of $\bar{\mathbf{p}}$ with respect to $\boldsymbol{\pi}$ and $\mathcal{W}(\bar{\mathbf{p}}')$ denotes the projection of $\mathcal{W}(\bar{\mathbf{p}})$ with respect to $\boldsymbol{\pi}$. $\bar{\mathbf{q}}'$ is the matching pixel of $\bar{\mathbf{q}}$ in the other view with respect to the plane $\boldsymbol{\pi}$. Let $\bar{\mathbf{q}} = (x, y)^{\intercal} \in I$. The disparity d of $\bar{\mathbf{q}}$ is given by eq. 2. Then $\bar{\mathbf{q}}' = (x - d, y)^{\intercal} \in I'$. If $\bar{\mathbf{q}} \in I'$, then $\bar{\mathbf{q}}' = (x + d, y)^{\intercal} \in I$.

The weight function $A(\bar{\mathbf{p}}, \bar{\mathbf{q}})$ in eq. 3 is used to overcome the edge fattening problem (described in Sec. II) and implements the adaptive support weight [25] by computing the affinity for $\bar{\mathbf{p}}$ and $\bar{\mathbf{q}}$ lying on the same plane by looking at the pixel's colour difference and spatial distance:

$$A(\bar{\mathbf{p}}, \bar{\mathbf{q}}) = \exp\left(-\frac{\Delta c_{\bar{\mathbf{p}}\bar{\mathbf{q}}}}{\gamma_{\mathrm{c}}}\right)\exp\left(-\frac{\Delta g_{\bar{\mathbf{p}}\bar{\mathbf{q}}}}{\gamma_{\mathrm{g}}}\right)\ , \tag{4}$$

where $\Delta c_{\bar{\mathbf{p}}\bar{\mathbf{q}}}$ measures the colour similarity and $\Delta g_{\bar{\mathbf{p}}\bar{\mathbf{q}}}$ denotes the geometric proximity between $\bar{\mathbf{q}}$ and the center pixel $\bar{\mathbf{p}}$ of $\mathcal{W}(\bar{\mathbf{p}})$. The user-defined parameters γ_{c} and γ_{g} are scale parameters and their values are discussed in Section VI. The colour similarity term $\Delta c_{\bar{\mathbf{p}}\bar{\mathbf{q}}}$ between $\bar{\mathbf{p}}$ and $\bar{\mathbf{q}}$ is defined as:

$$\Delta c_{\bar{\mathbf{p}}\bar{\mathbf{q}}} = ||I_{\bar{\mathbf{p}}} - I_{\bar{\mathbf{q}}}||\ ,$$

where $||I_{\bar{\mathbf{p}}} - I_{\bar{\mathbf{q}}}||$ computes the Euclidean distance of the colours of $\bar{\mathbf{p}}$ and $\bar{\mathbf{q}}$ in the CIELAB space, which approximates the perception of colour [25]. The geometric proximity term $\Delta g_{\bar{\mathbf{p}}\bar{\mathbf{q}}}$ is defined as the Euclidean distance between the coordinates of $\bar{\mathbf{p}}$ and $\bar{\mathbf{q}}$:

$$\Delta g_{\bar{\mathbf{p}}\bar{\mathbf{q}}} = ||\bar{\mathbf{p}} - \bar{\mathbf{q}}||\ .$$

The performance of a local stereo method depends to a large extent on the support weights that are used in the aggregation step [32]. It has also been shown in [32] that the adaptive support weight proposed in [25] works best in all considered scenarios including occluded regions [8]. The support weight $A(\bar{\mathbf{p}}', \bar{\mathbf{q}}')$ in the other view is also computed similarly.

The error function $E(\bar{\mathbf{q}}, \bar{\mathbf{q}}')$ is defined as the pixel dissimilarity between $\bar{\mathbf{q}}$ and $\bar{\mathbf{q}}'$:

$$E\big(\bar{\mathbf{q}}, \bar{\mathbf{q}}'\big) = (1-\alpha)\cdot\min\big(||I_{\bar{\mathbf{q}}}, I_{\bar{\mathbf{q}}'}||^2, \tau_{\mathrm{col}}\big) + \alpha\cdot\min\big(|\nabla I_{\bar{\mathbf{q}}} - \nabla I_{\bar{\mathbf{q}}'}|, \tau_{\mathrm{grad}}\big),$$

where $||I_{\bar{\mathbf{q}}}, I_{\bar{\mathbf{q}}'}||^2$ denotes the sum of squared distance (SSD) of $\bar{\mathbf{q}}$ and $\bar{\mathbf{q}}'$ in RGB [1] space and $|\nabla I_{\bar{\mathbf{q}}} - \nabla I_{\bar{\mathbf{q}}'}|$ denotes the absolute difference of grey-value gradients computed at $\bar{\mathbf{q}}$ and $\bar{\mathbf{q}}'$.

[1]We performed the same dissimilarity measure in CIELAB colour space but found that the RGB colour space produces better disparity map than CIELAB.

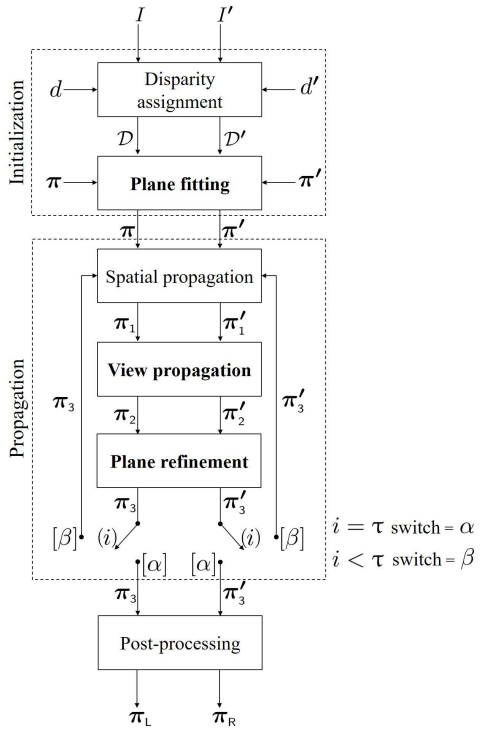

Fig. 3. Initialised PatchMatch Stereo (IPMS) flowchart. Input I and I' are the rectified reference and search images, respectively. Left and right disparity spaces $\mathcal{D}$ and $\mathcal{D}'$ are generated by selecting disparities d and d' from the disparity constraints for every pixel in I and I', respectively. The surface normals $\mathbf{n}$ and $\mathbf{n}'$ are selected from the normal constraints, which are used along with the point in the disparity space to generate the planes $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$ for every pixel in I and I', respectively. The total number of iterations is a user defined parameter τ and the iteration number is denoted by i. After each operation, the updated planes are represented by $\boldsymbol{\pi}_j$ and $\boldsymbol{\pi}'_j$. IPMS converges in two iterations in contrast to PMS, which takes three. After the post processing the final plane parameters for the left and right image are denoted by $\boldsymbol{\pi}_{\mathrm{L}}$ and $\boldsymbol{\pi}_{\mathrm{R}}$, respectively.

Since the x-coordinate of $\bar{\mathbf{q}}'$ lies in the continuous domain, we derive its colour and gradient values by linear interpolation. The user-defined parameter α balances the influence of the colour and gradient terms. Other user-defined parameters τ_{col} and τ_{grad} are thresholds that truncate costs for robustness in occluded regions. Their values are discussed in Section VI.

We updated the colour similarity term of IPMS to Truncated Sum of Squared intensity Differences (TSSD). Sum of Absolute intensity Differences (SAD) is more robust to noise and outliers than SSD, in the sense that the influence function is bounded [33]. However, both TSAD and TSSD are robust, as their influence functions are zero beyond the *outlier* threshold. Because of the sharp nature of SAD around the origin, SAD tends to lock on to a small number of very good matches in [8], which are then propagated. However, the smooth nature of SSD helps BOBYQA to improve the good matches more effectively than the original Luus-Jaakola scheme, as discussed in Section V. Additionally, when the noise is Gaussian [34], SSD is optimal for maximum-likelihood estimate, in the neighbourhood of a true match [35]. Experimental results in Fig. 8 also support this claim (see the results of IB and IBS).

C. Matching strategy

To assign the disparity at each pixel of I and I', we need to find a plane that minimises $\mathrm{cost}(\bar{\mathbf{p}}, \boldsymbol{\pi})$. IPMS follows the same framework of PMS with modifications highlighted in bold in Fig. 3.

We now discuss the three stages of the algorithm.

1) Initialisation: We use a uniform distribution that randomly assigns the disparities of each pixel in I and I' between the minimum allowed disparity, $d_{\min}$, and the maximum allowed disparity, $d_{\max}$:

$$d \sim U(d_{\min}, d_{\max}) \ . \tag{5}$$

This process hypothesis the left $(\mathcal{D})$ and right $(\mathcal{D}')$ disparity spaces. Then, we assign unit normal at each pixel of I and I' to find the plane parameters as discussed in Section IV. We use the same initialization process at each pixel of both images.

2) Propagation: We use the PMS iterative scheme to propagate the plane parameters in two different directions considering both views. In every iteration, each pixel runs through three independent stages: spatial propagation, view propagation, and plane refinement. We use the same *spatial propagation*[2] approach as mentioned in PMS (Fig. 4). As for *view propagation* , we exploit the strong coherency that exists between left and right disparity maps so that a pixel and its matching pixel in the other view have the same disparity. However, the surface normals in the disparity space change across views due to different view points. Let $\mathbf{n'}$ be the unit normal of the plane $\boldsymbol{\pi'}$ at $\mathbf{p'} \in \mathcal{D}'$. The relation between $\mathbf{n}$ and $\mathbf{n'}$ is:

$$\mathbf{n'} = \frac{\mathbf{M}^{-\intercal}\mathbf{n}}{||\mathbf{M}^{-\intercal}\mathbf{n}||} \ , \tag{6}$$

where $||\cdot||$ denotes the L_2 norm and $\mathbf{M}$ is defined in eq. 1.

In view propagation, we check all pixels of the second view that have our current pixel as a matching point according to their plane, and assign the plane parameters to the current pixel if the transformed plane in the first view reduces the cost.

Let $\{\bar{\mathbf{r}}, \bar{\mathbf{s}}\}$ be two possible matching points of $\bar{\mathbf{p}}$ in the other view. Let $\mathbf{r}$ and $\mathbf{s}$ be corresponding points of $\bar{\mathbf{r}}$ and $\bar{\mathbf{s}}$ in the disparity space lying on the plane $\boldsymbol{\pi}_\mathbf{r}$ and $\boldsymbol{\pi}_\mathbf{s}$ with disparity $d_\mathbf{r}$ and $d_\mathbf{s}$, and surface normals $\mathbf{n}_\mathbf{r}$ and $\mathbf{n}_\mathbf{s}$, respectively. Theoretically, matching pixels should have the same disparity magnitude with different unit normals. We transfer the normals $\mathbf{n}_\mathbf{r}$ and $\mathbf{n}_\mathbf{s}$ to the other view by eq. 6. Let $\mathbf{n}'_\mathbf{r}$ and $\mathbf{n}'_\mathbf{s}$ be the transformed unit normals of $\mathbf{n}_\mathbf{r}$ and $\mathbf{n}_\mathbf{s}$, and $\boldsymbol{\pi}'_\mathbf{r}$ and $\boldsymbol{\pi}'_\mathbf{s}$ be the their transformed planes respectively (Fig. 5). If the new plane parameters minimize the aggregated cost of $\mathcal{W}(\bar{\mathbf{p}})$ centred at $\bar{\mathbf{p}}$, then we update the plane parameters of $\bar{\mathbf{p}}$ with the

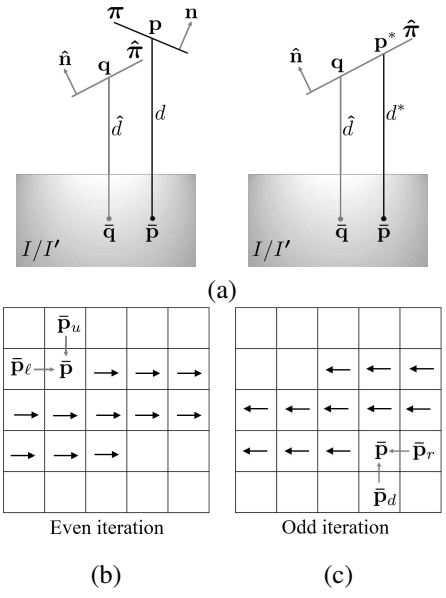

Fig. 4. Spatial propagation. (a) Image point $\bar{\mathbf{q}}$ is a spatial neighbour of $\bar{\mathbf{p}}$. Points $\mathbf{p}$ and $\mathbf{q}$ are corresponding points of $\bar{\mathbf{p}}$ and $\bar{\mathbf{q}}$ in the disparity space lying on the plane $\boldsymbol{\pi}$ and $\hat{\boldsymbol{\pi}}$ with unit normal $\mathbf{n}$ and $\hat{\mathbf{n}}$, and disparity d and $\hat{d}$, respectively. Image point $\bar{\mathbf{p}}$ corresponds to a new point $\mathbf{p}^*$ in the disparity space with respect to $\hat{\boldsymbol{\pi}}$. In spatial propagation we aggregate the cost of the patch $\mathcal{W}(\bar{\mathbf{p}})$ centred at $\bar{\mathbf{p}}$ with respect to $\boldsymbol{\pi}$ and $\hat{\boldsymbol{\pi}}$ if the new disparity d^* of $\bar{\mathbf{p}}$ with respect to $\hat{\boldsymbol{\pi}}$ is between $d_{\max}$ and $d_{\min}$. We update the plane of $\mathbf{p}$ to $\hat{\boldsymbol{\pi}}$ if the aggregated cost gets reduced by $\hat{\boldsymbol{\pi}}$. (b) and (c) show the direction of spatial propagation for odd and even iterations. The four immediate neighbours of $\bar{\mathbf{p}}$ are denoted by $\bar{\mathbf{p}}_\ell$, $\bar{\mathbf{p}}_r$, $\bar{\mathbf{p}}_u$, $\bar{\mathbf{p}}_d$ (left, right, upper and lower). In even iterations we consider the left and upper neighbours as spatial neighbours, whereas in odd iterations the right and lower neighbours are verified.

[2]During the experiments we observed that the horizontal neighbours dominate the spatial propagation. The vertical neighbours propagate less than 10% of the total.

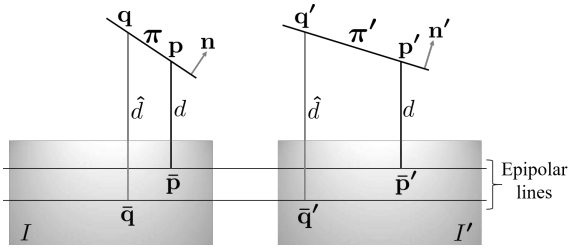

Fig. 5. Change of plane normals in view propagation. Image points $\bar{\mathbf{p}} \in I$ and $\bar{\mathbf{p}}' \in I'$ are two matching points. Points $\mathbf{p}$ and $\mathbf{p}'$ are corresponding points of $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ in the disparity space lying on the plane $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$ with disparity d, and surface normal $\mathbf{n}$ and $\mathbf{n}'$, respectively. Let $\bar{\mathbf{q}}$ be a neighbouring point of $\bar{\mathbf{p}}$ and $\bar{\mathbf{q}}'$ be the matching pixel of $\bar{\mathbf{q}}$ in the other view. By transforming the plane normals in the other view, we can show that the disparity $\hat{d}$ of $\bar{\mathbf{q}}$ and $\bar{\mathbf{q}}'$ are equal with respect to $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$, respectively. In the figure, $\mathbf{q}$ and $\mathbf{q}'$ are corresponding points of $\bar{\mathbf{q}}$ of $\bar{\mathbf{q}}'$ in the disparity space with respect to $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$.

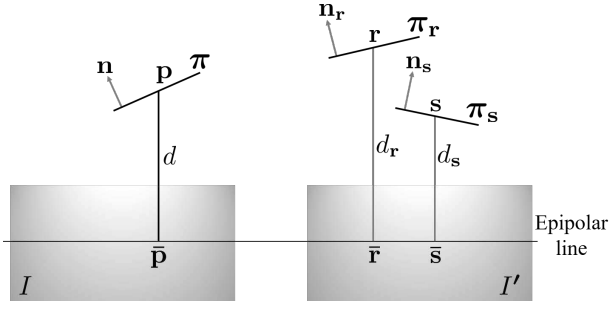

Fig. 6. View propagation. Let $\{\bar{\mathbf{r}}, \bar{\mathbf{s}}\} \in I'$ are two possible matching points of $\bar{\mathbf{p}} \in I$. Points $\mathbf{p}$, $\mathbf{r}$ and $\mathbf{s}$ are corresponding points of $\bar{\mathbf{p}}$, $\bar{\mathbf{r}}$, $\bar{\mathbf{s}}$ in the disparity space lying on the plane $\boldsymbol{\pi}$, $\boldsymbol{\pi}_\mathbf{r}$ and $\boldsymbol{\pi}_\mathbf{s}$ with disparity d, $d_\mathbf{r}$ and $d_\mathbf{s}$, and surface normal $\mathbf{n}$, $\mathbf{n}_\mathbf{r}$ and $\mathbf{n}_\mathbf{s}$, respectively. We transfer the normal and disparity of $\bar{\mathbf{r}}$ and $\bar{\mathbf{s}}$ to $\bar{\mathbf{p}}$ and find the new plane parameters. If the new plane parameters minimize the aggregated cost of the patch $\mathcal{W}(\bar{\mathbf{p}})$ centred at $\bar{\mathbf{p}}$, we update the plane parameters of $\bar{\mathbf{p}}$ with the new one.

new values (Fig. 6). Due to rectification error, corresponding points may be vertically displaced by one or two pixels. In our modified view propagation, we address this problem by assigning plane parameters to the immediate neighbours of the candidate pixels. We assign the disparity and the transformed plane normal of $\bar{\mathbf{r}}$ and $\bar{\mathbf{s}}$ to the four immediate neighbours (left, right, upper, lower) of $\bar{\mathbf{p}}$ and check whether the new plane parameters reduce the cost. If so, we also update the planes of the neighbours accordingly.

Finally, in *plane refinement*, we use a trust region based, gradient free non-linear optimiser BOBYQA [15] to further refine the plane parameters, as discussed in Section V.

3) Post-processing: We follow the same post-processing scheme as mentioned in the PMS [8]. Additionally, we apply a median filter on each pixel of the resulting disparity map to eliminate any isolated mismatches [36].

IV. Constrained Plane Initialization

The disparity range assumption states that the disparity of any pixel inside an image should lie between the maximum and minimum disparity. The disparity of a pixel within a patch is computed with respect to the plane associated with the centre pixel of the patch. Hence it is important to assign the plane normals in a suitable way such that they comply with the disparity range assumption. Heise *et al.* [9] tried to solve the initialization problem by constraining the first two components of the plane normal equally distributed over a circle (within 0.5 radius) in the disparity space. Galliani *et al.* [11] generated the unit normals equally distributed over the sphere in the disparity space. Both these strategies may associate infeasible planes to some pixels. Moreover, [9] does not give any justification for choosing the radius of 0.5.

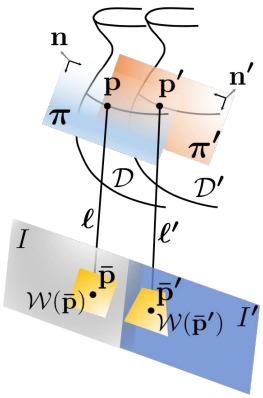

Fig. 7. Surface normal constraints. Image points $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ are matching pixels with disparity d lying on the reference image I and the search image I', respectively. The matching pixels $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ correspond to $\mathbf{p} \in \mathcal{D}$ and $\mathbf{p}' \in \mathcal{D}'$, where $\mathcal{D}$ and $\mathcal{D}'$ represent the disparity spaces generated by I and I', respectively. Points $\mathbf{p}$ and $\mathbf{p}'$ lie on the plane $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$ with unit surface normal $\mathbf{n}$ and $\mathbf{n}'$ in the outward direction, respectively. A rectangular patch centred at $\bar{\mathbf{p}}$ is denoted by $\mathcal{W}(\bar{\mathbf{p}})$. The patch $\mathcal{W}(\bar{\mathbf{p}}')$ is the projection of $\mathcal{W}(\bar{\mathbf{p}})$ with respect to $\boldsymbol{\pi}$. The vector joining $\mathbf{p}$ and $\bar{\mathbf{p}}$ is defined as the line of sight vector $\boldsymbol{\ell}$ with respect to the left camera coordinate system. Similarly, $\boldsymbol{\ell}'$ is the line of sight vector joining $\mathbf{p}'$ and $\bar{\mathbf{p}}'$ with respect to the right camera coordinate system. The visibility constraint in the disparity space assumes $\boldsymbol{\pi}$ is visible from $\boldsymbol{\ell}$ and $\boldsymbol{\pi}'$ is visible from $\boldsymbol{\ell}'$. The disparity bound constraint on support window assumes the disparity of all the pixels inside $\mathcal{W}(\bar{\mathbf{p}})$ with respect to $\boldsymbol{\pi}$ is between $d_{\min}$ and $d_{\max}$.

The core idea behind our constrained initialization is to only assign geometrically feasible planes to every pixel of both images during initialization. Furthermore, the plane normal bounds are maintained later during the plane refinement scheme. There can be no disadvantage with respect to PMS, because infeasible matches should never be propagated. The initialization scheme is an important part of the framework as the rate of convergence depends on the assignment of correct plane parameters during initialization. The more viably the plane parameters are estimated, the faster the algorithm converges. The positive effects of the constrained initialization are demonstrated during experiments (Fig. 11).

A. Visibility constraint in the disparity space

When searching for the optimal plane containing a given scene point, we only consider those planes which are visible from the camera at the scene point [37]. We apply this constraint in the disparity space. The visibility constraint in the disparity space assumes that the plane in the disparity space is visible from the line of sight vector joining the image point and the corresponding point in the disparity space (Fig. 7). The constraint only considers whether the center point of the patch and its corresponding point in the other view is visible from individual cameras. Due to various patch orientations there may be cases where the patch is partially visible from the other camera. This issue is taken care by the truncation parameters in the cost function (eq. 3). This visibility constraint is only a necessary constraint that differs from the general visibility constraint that checks whether the surface points are truly visible in two views without occlusion. The general visibility constraint can be made precise in terms of disparity gradient.

If $\bar{\mathbf{p}} = (\xi, \eta) \in I$ with disparity δ, then the matching pixel in the other view is $\bar{\mathbf{p}}' = (\xi - \delta, \eta)^{\mathsf{T}} \in I'$ (Fig. 7). The corresponding points of $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ in the disparity space are $\mathbf{p} = (\xi, \eta, \delta)^{\mathsf{T}} \in \mathcal{D}$ and $\mathbf{p}' = (\xi - \delta, \eta, \delta)^{\mathsf{T}} \in \mathcal{D}'$, respectively. The unit surface normal in I' is given by eq. 6.

In the left camera coordinate system, the line of sight vector of $\mathbf{n}$ at $\mathbf{p}$ with respect to $\bar{\mathbf{p}}$ is $\boldsymbol{\ell} = (0, 0, \delta)^{\mathsf{T}}$. Similarly, in the right camera coordinate system, the line of sight vector of $\mathbf{n}'$ at $\mathbf{p}'$ with respect to $\bar{\mathbf{p}}'$ is $\boldsymbol{\ell}' = (0, 0, \delta)^{\mathsf{T}}$. Planes $\boldsymbol{\pi}$ and $\boldsymbol{\pi}'$ are visible from $\boldsymbol{\ell}$ and $\boldsymbol{\ell}'$, respectively, if

$$\mathbf{n} \cdot \boldsymbol{\ell} > 0 \quad \text{and} \quad \mathbf{n}' \cdot \boldsymbol{\ell}' > 0 \; . \tag{7}$$

Substituting $\mathbf{n}$ and $\mathbf{n}'$ by their corresponding normal components u, v and w in inequality 7 we get

$$w > 0 \quad \text{and} \quad u > -w \; . \tag{8}$$

If $\bar{\mathbf{p}} \in I'$, then its matching pixel $\bar{\mathbf{p}}' = (\xi + \delta, \eta)^{\mathsf{T}} \in I$. The corresponding points of $\bar{\mathbf{p}}$ and $\bar{\mathbf{p}}'$ in the disparity space are $\mathbf{p} = (\xi, \eta, \delta)^{\mathsf{T}} \in \mathcal{D}'$ and $\mathbf{p}' = (\xi + \delta, \eta, \delta)^{\mathsf{T}} \in \mathcal{D}$, respectively. In both camera coordinate systems the line of sight direction remains unchanged except the plane normal that can be retrieved using eq. 6. Again from inequality 7 we get:

$$w > 0 \quad \text{and} \quad u < w \; . \tag{9}$$

Both inequality 8 and 9 imply that w is positive. We also get a bound (upper or lower) on u depending on which image $\bar{\mathbf{p}}$ is

taken from. The visibility constraint does not give a complete bound (both upper and lower) for u. Moreover, the constraint does not provide any bounds for v. Therefore, we introduce our second constraint to obtain complete bounds on u and v.

B. Disparity bound constraint on support window

From the cost function (eq. 3), we know that the algorithm projects a support window in the other view by eq. 2 and then compare the weighted pixel difference between the two support windows to measure the matching score. To find the complete bound on the plane normals, our disparity bound constraint on the support window assumes that the disparity of all the pixels inside a support window with respect to the plane associated with the centre pixel of the patch must lie between the maximum and the minimum allowed disparity, *i.e.,* the disparity of every pixel $\bar{\mathbf{q}} = (x, y) \in \mathcal{W}(\bar{\mathbf{p}})$ with respect to $\boldsymbol{\pi}$ should lie between $d_{\max}$ and $d_{\min}$ (Fig. 7). We should only consider planes that satisfy the following disparity bound condition:

$$d_{\min} \leq |\boldsymbol{\pi} \cdot \bar{\mathbf{q}}| \leq d_{\max} \quad \forall\ \bar{\mathbf{q}} \in \mathcal{W}(\bar{\mathbf{p}}),$$

where $|\boldsymbol{\pi} \cdot \bar{\mathbf{q}}|$ computes the perpendicular distance of $\bar{\mathbf{q}}$ to $\boldsymbol{\pi}$. This distance gives the disparity of $\bar{\mathbf{q}}$ with respect to $\boldsymbol{\pi}$ at $\mathbf{p}$. From eq. 2, the disparity d of $\bar{\mathbf{q}}$ is given by

$$d = \frac{u}{w}(\xi - x) + \frac{v}{w}(\eta - y) + \delta \ .$$

The disparity of $\bar{\mathbf{q}}$ should also satisfy the disparity bound condition:

$$d_{\min} \leq \frac{u}{w}(\xi - x) + \frac{v}{w}(\eta - y) + \delta \leq d_{\max} \ . \tag{10}$$

From the visibility constraint, we know that the value of w is always positive. Then inequality 10 can be simplified as:

$$-w\,(\delta - d_{\min}) \leq u\,(\xi - x) + v\,(\eta - y) \leq w\,(d_{\max} - \delta) \ . \tag{11}$$

Both $(\delta - d_{\min})$ and $(d_{\max} - \delta)$ are positive in the above expression. If $d^* = \min\,(\delta - d_{\min}, d_{\max} - \delta)$, we tighten the bounds of inequality 11 as:

$$-wd^* \leq u\,(\xi - x) + v\,(\eta - y) \leq wd^* \ . \tag{12}$$

Any solutions of inequality 12 will also satisfy inequality 11. The values of $(\xi - x)$ and $(\eta - y)$ depend on the patch size. For a patch of size $2r + 1$, both their values ranges from $-r$ to r. A solution of inequality 12 for $\pm r$ is also valid for other values $< |r|$. Therefore, a solution of inequality 12 will also satisfy the following inequalities:

$$-wd^* \leq ur \pm vr \leq wd^* \ . \tag{13}$$

We can now solve inequality 13 to get the upper and lower bound of u.

$$-\frac{wd^*}{r} \leq u \leq \frac{wd^*}{r} \ . \tag{14}$$

A geometrical formulation on the bounds of u/w is shown in Fig. 8. Applying similar disparity bound constraint for the other view, we find the analogous representation of inequality 13 as:

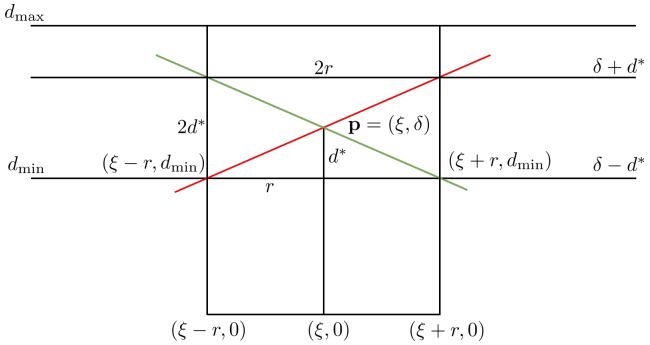

Fig. 8. Bounds on u/w. Let $d^* = \delta - d_{\min}$. In the uw plane both the red and green lines (originated from $\mathbf{p}$) are extreme lines that follow the disparity bound constraint. Slopes of the red and the green line are d^*/r and $-d^*/r$, respectively. Any line whose slope is between $-d^*/r$ and d^*/r is a potential candidate plane.

$$-d^* \leq \frac{u}{w+u}r \pm \frac{v}{w+u}r \leq d^* \ . \tag{15}$$

If $\bar{\mathbf{p}} \in I'$, the equivalent representation of inequality 13 for the other view changes to:

$$-d^* \leq \frac{u}{w-u}r \pm \frac{v}{w-u}r \leq d^* \ . \tag{16}$$

Combining the bounds of u from inequality 14, 15 and 16 with the bounds from the visibility constraint (inequality 8 and 9), we get the bounds for u satisfying both constraints:

$$\begin{aligned} -\frac{wd^*}{r+d^*} < u \leq \frac{wd^*}{r} \quad &\text{if } \bar{\mathbf{p}} \in I \ , \\ -\frac{wd^*}{r} \leq u < \frac{wd^*}{r+d^*} \quad &\text{if } \bar{\mathbf{p}} \in I' \ , \end{aligned} \tag{17}$$

The value of v is constrained by tightening the bounds and using inequality 13, 15 and 16.

$$v \in \frac{s}{r}\,[-1, 1], \tag{18}$$

where

$$s = \begin{cases} \min\big(|wd^* \pm ur|, |(w+u)d^* \pm ur|\big) & \text{if } \bar{\mathbf{p}} \in I \ , \\ \min\big(|wd^* \pm ur|, |(w-u)d^* \pm ur|\big) & \text{if } \bar{\mathbf{p}} \in I' \ . \end{cases}$$

Both the constraints are hard constraints used during the initialization and the plane refinement process. The proposed scheme samples the unit normals multiple times, until the constraints are satisfied. Using [38], we first generate random unit normals that are uniformly distributed over the visible hemisphere. Next, we only accept those unit normals that satisfy the constraints in inequality 8, 9, 17 and 18. The fronto-parallel windows can be enforced by setting $\mathbf{n} = (0, 0, 1)^\intercal$. Then we use d and $\mathbf{n}$ to find the required plane parameters at $\mathbf{p}$. These plane parameters are then used in eq. 3 to find the disparity of all the other pixels inside a patch. During initialization and plane refinement, while minimising the cost function over disparity, our constraints ensure that the center pixel of the patch is visible from both views and the disparity of all the pixels inside a patch lie between the minimum and maximum disparity.

V. Constrained optimization

We finally refine the plane parameters of $\boldsymbol{\pi}$ at a pixel $\bar{\mathbf{p}}$ to further reduce the matching cost. Here we change the disparity and the surface normal within bounds and seek an optimum disparity and unit normal. As our cost function is non-differentiable at some points due to the presence of discontinuous thresholds in the pixel dissimilarity function, we cannot use any standard gradient descent method to minimize it. It is also not convenient to mathematically compute the derivatives. PMS uses a Luus-Jaakola type scheme which is not always effective as it uses a heuristic approach to find the ideal plane parameters instead of using the local structure of the cost function.

BOBYQA [15] [16] is a gradient free non-linear trust region based algorithm for finding the minimum of an objective function $F(\mathbf{x})$, $\mathbf{x} \in \mathbb{R}^n$, subject to some constraints $\mathbf{a} \leq \mathbf{x} \leq \mathbf{b}$, without using the derivatives of $F(\mathbf{x})$. A trust region is a neighbourhood of the current iterate point which is used in conjunction with a local quadratic approximation Q of F at that point. The quadratic surface Q is then minimised and if the minimum of Q also reduces the value of F, BOBYQA treats the minimum as a new iterate point and generates a bigger trust region around the minimum and continues to search for new minima. Otherwise, a smaller trust region is generated around the iterate point. The radius of the trust region depends on how well the quadratic model matches with the objective function and is updated after each iteration. BOBYQA consists of a very accurate and efficient system of updating the approximation models and it maintains a good set of interpolation points.

BOBYQA starts with an initial vector $\mathbf{x}$ of dimension n, the constraints on $\mathbf{x}$ and a trust region radius. In each iteration, BOBYQA employs a local quadratic approximation Q of F such that $Q(\mathbf{x}_j) = F(\mathbf{x}_j)$, $j = 1, 2, \cdots, m$, where $m = 2n + 1$. The initial interpolation points $\mathbf{x}_j$ are chosen and adjusted automatically. Let $\mathbf{x}_k$ be the point in the set $\{\mathbf{x}_j : j = 1, 2, \cdots, m\}$ that has the property $F(\mathbf{x}_k) = \min\{F(\mathbf{x}_j) : j = 1, 2, ..., m\}$ associated with the current trust region radius Δ_k. At each iteration, a new point $\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{c}_k$, $\mathbf{c}_k \in \mathbb{R}^n$ is computed and one of the interpolation points, say $\mathbf{x}_j$, is replaced by the new iterate point $\mathbf{x}_{k+1}$ if $F(\mathbf{x}_{k+1}) < F(\mathbf{x}_k)$. The trust region step $\mathbf{c}_k$ is chosen by minimising $Q(\mathbf{x}_k + \mathbf{c})$, $\mathbf{c} \in \mathbb{R}^n$ subject to the prescribed bounds on variables $\mathbf{a} \leq \mathbf{x}_k + \mathbf{c} \leq \mathbf{b}$ under the condition $||\mathbf{c}|| \leq \Delta_k$. Further, a new trust region radius and quadratic approximation is generated using the new iterate point. In most cases the new trust region is computed as $\max\left(\frac{1}{2}\Delta_k, \|\mathbf{c}_k\|\right)$. Thus at each iteration, only one interpolation point is altered that minimises F among all the interpolation points from the minimising sequence $\mathbf{x}_k^\star$.

The disparity and the plane normals of a pixel obtained from the view propagation are used as initial inputs. The optimiser then minimises $\mathrm{cost}(\bar{\mathbf{p}}, \boldsymbol{\pi})$ using the disparity and surface normal bounds as defined in Section III-C1 and IV. The disparity scale is very different from the unit normal scale, but BOBYQA compensates for unequal initial-step sizes in the different parameters by rescaling the parameters proportional to the initial trust region step [39]. As the input and the boundary conditions vary for each pixel, we let the BOBYQA subroutine compute the initial trust region radius heuristically from the bounds. The input vector in our case is the over-parametrized plane parameters of dimension $n = 4$ containing the disparity and the plane normals of a point in the disparity space. BOBYQA then heuristically interpolates $m = 9$ planes with the bounds and fits a quadratic approximation Q around the $\mathrm{cost}(\bar{\mathbf{p}}, \boldsymbol{\pi})$. The quadratic model is later used to minimize the cost function along the trust region step $\mathbf{c}_k$.

VI. Results

A. Experimental setup

We evaluate our proposed method, IPMS, using the Middlebury stereo benchmark, version 3 [21], which contains a training and a test dataset, each with 15 pairs of stereo rectified images. The images contain a variety of challenges, such as radiometric changes and a large disparity range. Most images in the dataset have realistic *imperfect* rectification. The dataset comes in three resolutions (full, half and quarter). Due to the computational memory constraint, we use the half resolution dataset (up to 1500×1000 pixels) for our experiments. However, results are evaluated at full resolution to compare with other methods according to the benchmark.

There are six parameters in IPMS. We use the same parameters for all the stereo pairs. All the parameters are set empirically during training. We keep the patch size $\mathcal{W}(\bar{\mathbf{p}}) = 71 \times 71$ and use a 5×5 median filter for all the half resolution Middlebury images. The results are reported after the post processing step, unless otherwise stated. The values of γ_{c} and γ_{g} are the same as used in [25], α and τ_{grad} are the same as in [8] and τ_{col} is chosen empirically using the quarter resolution Middlebury training dataset:

$$\left(\gamma_{\mathrm{c}},\ \gamma_{\mathrm{g}},\ \alpha,\ \tau_{\mathrm{col}},\ \tau_{\mathrm{grad}}\right) := \left(5.0,\ \frac{\mathcal{W}(\bar{\mathbf{p}})}{2},\ 0.9,\ 0.01,\ 0.008\right).$$

As a performance measure, we use the default metrics of the Middlebury stereo benchmark. The Middlebury error rate measures the *percentage of bad pixels*, *i.e.,* the percentage of pixels whose disparity errors are greater than a threshold with respect to the ground-truth disparity map. The default metric also measures the *average error* per pixel. The training dataset provides a mask for the occluded pixels allowing users to calculate the percentage of bad pixels and average error only on the non-occluded pixels. The percentage of bad pixels changes with the threshold, whereas the average error remains constant. We use the thresholds of 2.0 and 0.5 pixels.

B. Comparison with PMS

Let us first compare our results with PMS and other variants of PMS on a subset of the Middlebury training images. This subset was chosen because it exhibits different scene textures and disparity ranges. Results in Fig. 9 clearly show that the proposed modifications significantly decrease the percentage of bad pixels by $10 - 40\%$ and the average error by $25 - 50\%$ in most cases. It is also evident from Fig. 9 that the BOBYQA optimization is more effective than the Luus-Jaakola method.

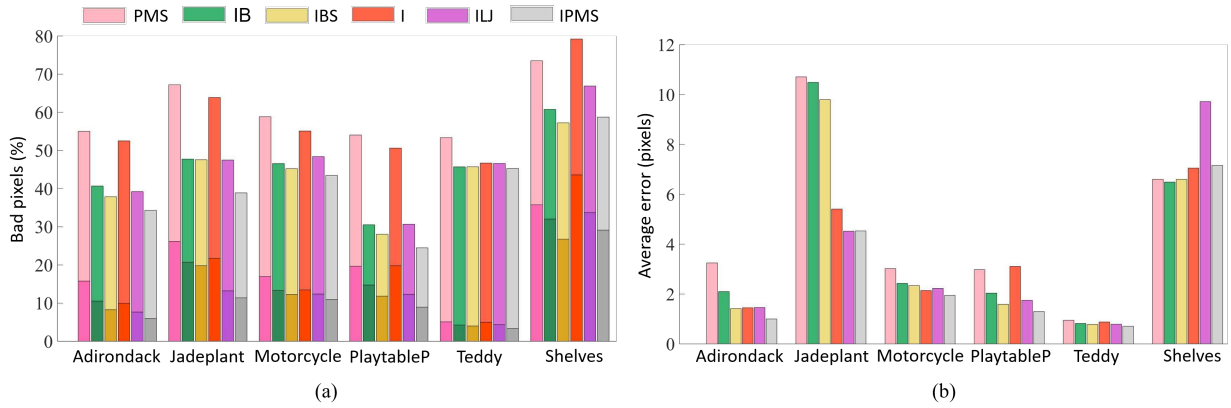

Fig. 9. Error comparison on a subset of the Middlebury training images (`Adirondack`, `Jadeplant`, `Motorcycle`, `PlaytableP`, `Teddy`, and `Vintage`). PMS: **P**atch**M**atch **S**tereo [8]; IB: PatchMatch Stereo with constrained **I**nitialisation and **B**OBYQA in plane refinement, cost function same as [8], no search image support weight; IBS: PMS with constrained **I**nitialisation and **B**OBYQA in plane refinement, cost function similar to [8], norm changed by TS**S**D, no search image support weight; I: **I**nitialised PatchMatch stereo with no plane refinement; ILJ : **I**nitialised PatchMatch stereo with a variant of **L**uus-**J**aakola optimization in plane refinement; IPMS : **I**nitialised **P**atch**M**atch **S**tereo. (a) Dark and light shades represent percentage of non occluded bad pixels computed with error threshold of 2.0 and 0.5 pixels, respectively. (b) Average disparity error for non occluded pixels.

TABLE II
PERCENTAGE OF BAD PIXELS WITH 2.0 PIXELS ERROR THRESHOLD ON ALL PIXELS ON A SUBSET OF THE MIDDLEBURY HALF-RESOLUTION TRAINING DATASET. THE MEAN ERROR IS REPORTED FOR 30 TRIALS ALONG WITH THE STANDARD DEVIATION.

Method	weights		1	1	1	1	0.5	1
	mean	median	`Adiron`	`Jadepl`	`Motor`	`PlaytP`	`Shelvs`	`Teddy`
IPMS SP1	54.19	56.31	48.64 (1.55)	73.13 (0.81)	48.82 (1.47)	63.81 (2.17)	65.41 (0.79)	30.95 (1.88)
IPMS SP1 with init. as [11]	67.56	67.42	66.47 (1.63)	81.26 (0.78)	65.36 (1.24)	68.37 (1.69)	75.99 (1.15)	52.16 (1.87)
IPMS VP1	39.26	39.38	30.94 (1.19)	56.82 (0.58)	34.24 (0.56)	44.53 (1.48)	58.81 (0.37)	20.02 (0.59)
IPMS VP1 no neighbour	47.53	47.57	41.36 (1.08)	67.66 (1.08)	42.91 (0.99)	52.23 (0.76)	63.45 (0.73)	25.56 (0.55)
IPMS PR1	35.95	36.09	26.29 (1.09)	52.07 (0.62)	31.65 (0.49)	40.53 (1.35)	57.27 (0.34)	18.57 (0.47)
IMPS PR1 no vis. const. in opt.	39.85	39.01	33.61 (0.76)	56.28 (0.99)	35.27 (0.41)	42.73 (0.85)	60.84 (0.36)	20.84 (0.39)
IPMS	17.92	14.77	9.65 (0.25)	25.81 (0.37)	15.93 (0.48)	13.62 (0.42)	50.18 (0.35)	8.48 (0.27)
IPMS with init. as [11]	19.09	16.11	9.88 (0.28)	27.79 (0.43)	17.04 (0.50)	15.18 (0.36)	51.53 (0.56)	9.39 (0.47)
IPMS no neighbour in VP	18.63	15.57	9.81 (0.23)	27.41 (0.36)	16.86 (0.33)	14.28 (0.27)	51.09 (0.37)	8.56 (0.41)
IMPS no vis. const. in opt.	18.66	15.61	10.03 (0.19)	27.29 (0.36)	16.93 (0.44)	14.29 (0.48)	51.04 (0.32)	8.58 (0.35)

We compared both the initialization schemes of IPMS and PMS and found that IPMS produces $35-45\%$ fewer bad planes than that of PMS during initialization. We also compared the initialization scheme of IPMS with that by Galliani *et al.* [11], which generates uniform normals on a sphere. Experimental results show that the proposed IPMS produces $30-40\%$ fewer bad planes compared to [11] during initialization, $11-37\%$ fewer bad pixels during the first spatial propagation and $3-11\%$ fewer bad pixels on the final disparity map for error threshold of 2.0 on all pixels (Table II).

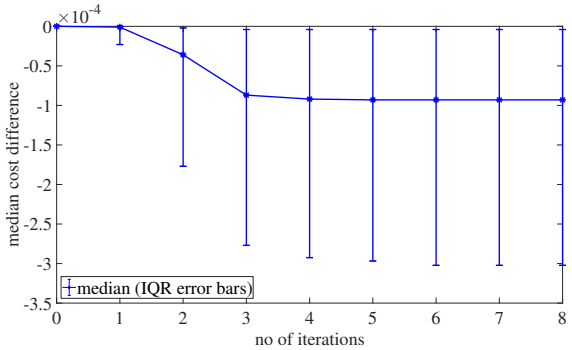

Fig. 10. Cost difference of BOBYQA and Luus-Jaakola on the medium size `Jadeplant` dataset. We choose a grid of equally spaced points (131×99), 10 pixels apart and plot the median cost difference of BOBYQA and Luus-Jaakola along with the upper and lower quartile with respect to the no. of iterations. Both the optimisers start off from the same median cost but after the first iteration BOBYQA proves to be better than Luus-Jaakola.

Our modified view propagation allows more opportunities to improve the disparity by testing with neighbouring parameters. This approach is particularly useful where corresponding points are off by one or two pixels due to imperfect rectification. We performed new experiments on the Playtable stereo pair with both imperfect and perfect rectification [21] (Table III). The results show that, in the case of imperfect

TABLE III
PERCENTAGE OF BAD PIXELS WITH 2.0 ERROR THRESHOLD ON ALL PIXELS FOR THE `Playtable` STEREO PAIR WITH PERFECT AND IMPERFECT RECTIFICATION IN THE FIRST (FVP) AND SECOND (SVP) VIEW PROPAGATION ALONG WITH THE FINAL DISPARITY DISPARITY MAP. THE MEAN ERROR IS REPORTED FOR 30 TRIALS ALONG WITH THE STANDARD DEVIATION.

Rectification	Disparity results	No neighbour	With neighbour
Imperfect	FVP	53.45 (2.01)	48.46 (1.87)
	SVP	40.12 (1.21)	31.34 (1.03)
	Final	**33.37 (0.96)**	**26.73 (0.76)**
Perfect	FVP	43.45 (1.42)	44.53 (1.48)
	SVP	19.33 (0.99)	19.12 (0.85)
	Final	**14.28 (0.37)**	**13.62 (0.42)**

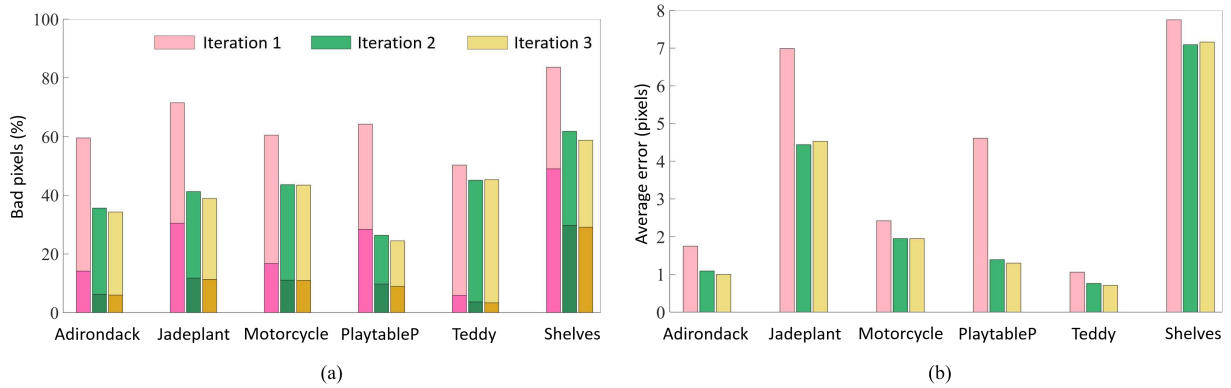

Fig. 11. Rate of convergence per iteration computed on a subset of the Middlebury training images (`Adirondack`, `Jadeplant`, `Motorcycle`, `PlaytableP`, `Teddy`, and `Vintage`). Note that two iterations are sufficient for convergence. (a) Dark and light shades represent percentage of non occluded bad pixels computed with error threshold of 2.0 and 0.5 pixels, respectively. (b) Average disparity error for non occluded pixels.

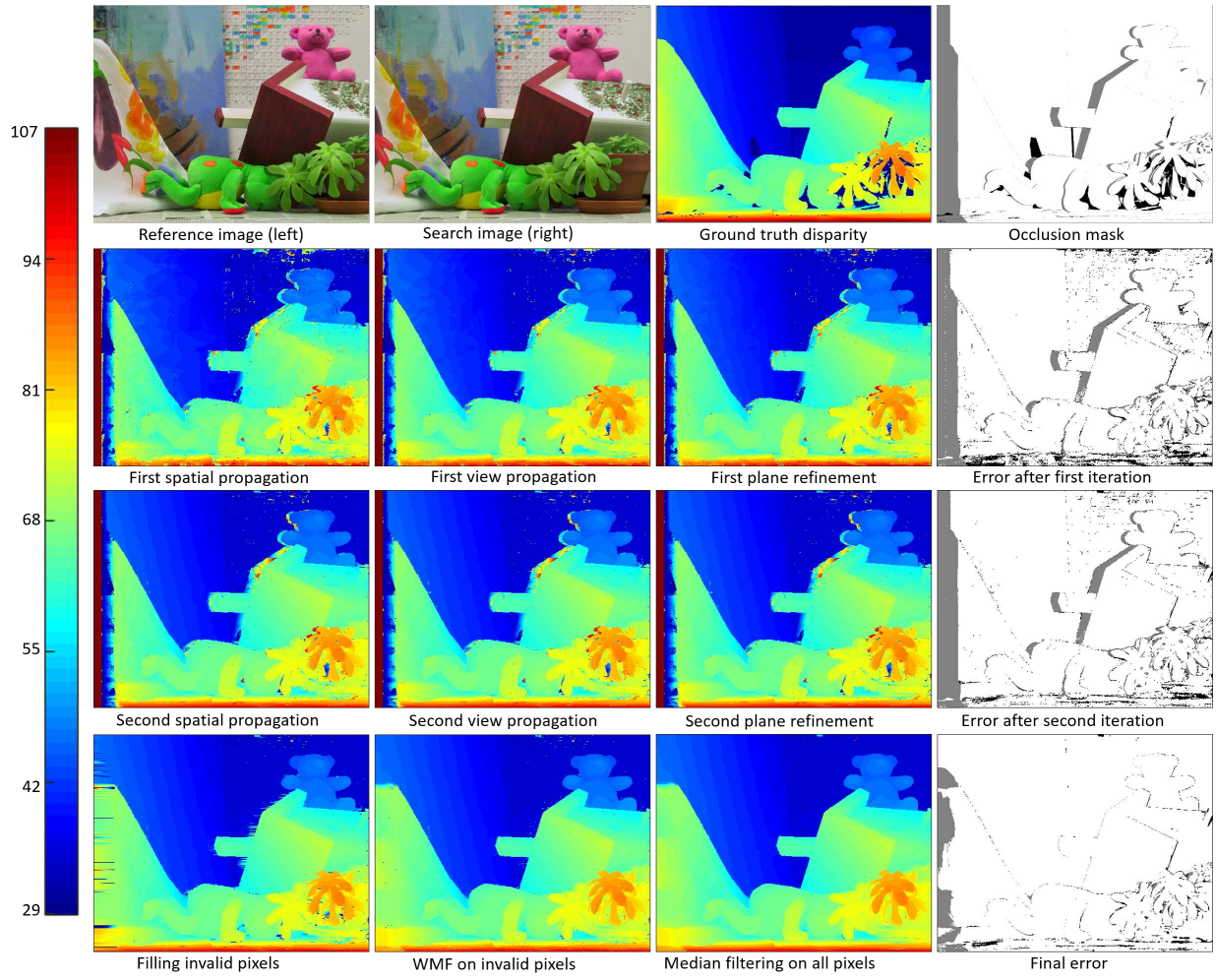

Fig. 12. Teddy stereo pair results with default error (percentage of bad pixels) threshold of 2.0 pixels on all regions. Errors in occluded regions in the right-most column are shown in grey, while the errors in non-occluded areas are shown in black. The colour bar in the left-most column shows the disparity levels of the Teddy stereo pair.

rectification, there is a 20% improvement on the final disparity map. In comparison, for the perfectly rectified case, use of the full neighbourhood in view propagation did not change the final disparity results. Overall, the modified view propagation results in $20-35\%$ more plane propagations, reducing $17-28\%$ bad pixels during the first view propagation and $2-6\%$ fewer bad pixels on the final disparity map for error threshold of 2.0 on all pixels.

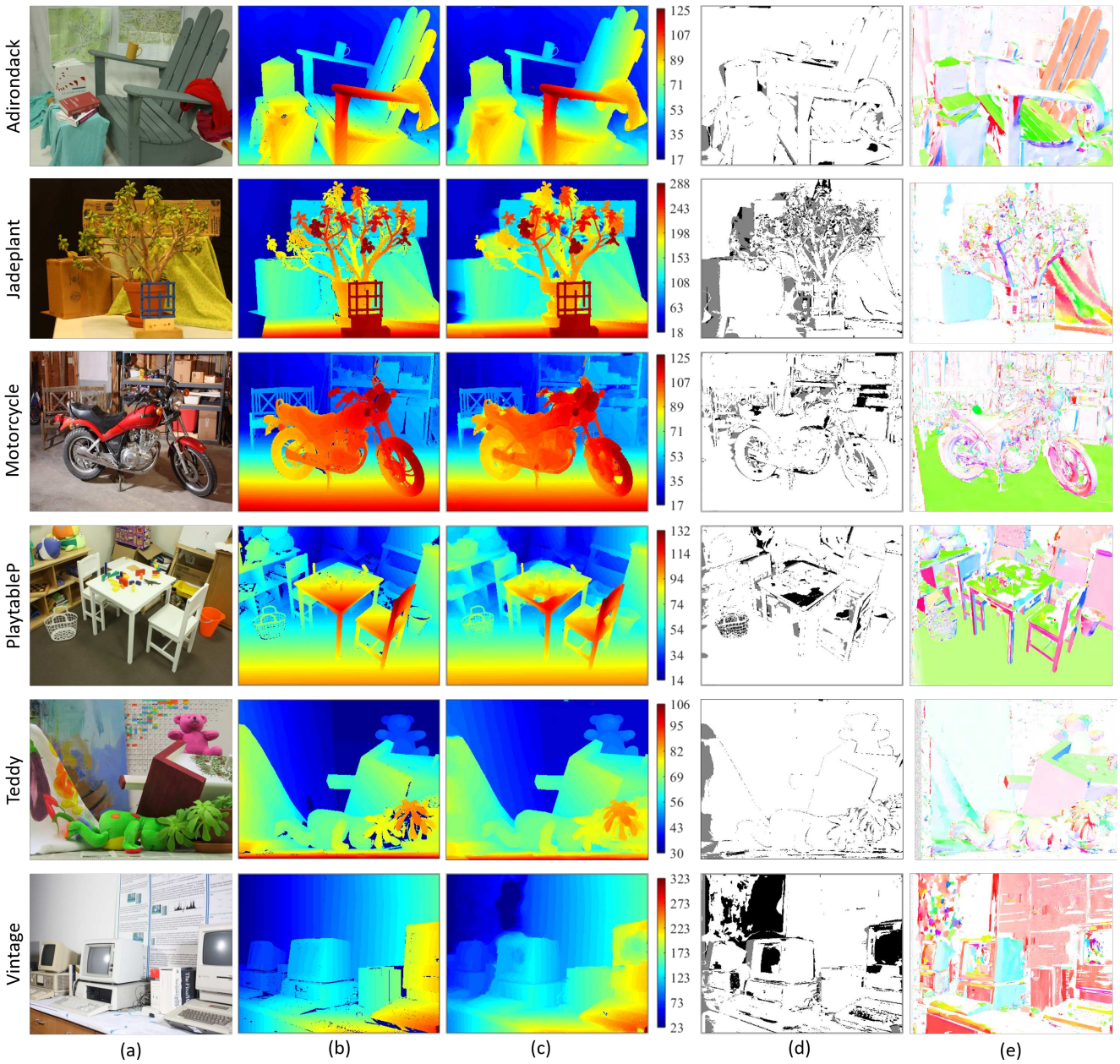

Fig. 13. Results on a subset of the (a) Middlebury training images with (b) ground truth (`Adirondack`, `Jadeplant`, `Motorcycle`, `PlaytableP`, `Teddy`, and `Vintage`). (c) Disparity map. (d) Error map. Computed with default error threshold of 2.0 pixels on all regions. Errors in occluded regions are shown in grey, while the errors in non-occluded areas are shown in black. (e) Plane distribution. Hue and saturation represent surface tilt and slant, with white being fronto-parallel.

We also performed an experiment to find out the effectiveness of the constraints during optimization. The performance of the BOBYQA optimiser depends on the specified bounds of the input variables. In the absence of visibility constraints, the trust region for the normal parameters will be unnecessarily large. This may result in a poor quadratic approximation to the empirical objective function, such that the minimum of the approximation does not offer any improvement (e.g. consider a U-shaped approximation around a W-shaped function). This interpretation is consistent with our experimental results in Table II. In addition, it may be noted that although the disparity scale is very different from the unit normal scale, BOBYQA rescales the parameters in relation to the initial trust region step [39]. This process also depends on the specified bounds. Experimental results show $2-10\%$ fewer bad pixels during the first plane refinement and $2-6\%$ fewer bad pixels on the final disparity map for error threshold of 2.0 on all pixels.

We choose a sample of equally spaced pixels (131×99) on the half resolution `Jadeplant` stereo pair (10 pixels apart) and compared the median cost difference of BOBYQA and Luus-Jaakola along with the upper and lower quartile with respect to the number of iterations (Fig. 10). Both the optimisers start from the same median cost, but after the first iteration BOBYQA proves to be better than Luus-Jaakola.

TABLE IV
RESULTS ON THE MIDDLEBURY TRAINING SET: PERCENTAGE OF BAD PIXELS AND AVERAGE ERROR WITH 2.0 PIXEL ERROR THRESHOLD ON ALL PIXELS

		Method	Weights →	1	1	1	1	1	1	0.5	1	0.5	0.5	1	1	0.5	1	0.5
			Wt. avg.	Adiron	ArtL	Jadepl	Motor	MotorE	Piano	PianoL	Pipes	Playrm	Playt	PlaytP	Recyc	Shelvs	Teddy	Vintge
Bad pixel (%)	Global	MC-CNN-acrt [40]	19.9	10.0	25.6	32.8	12.7	12.6	18.8	24.4	**17.8**	25.6	23.3	21.6	**12.8**	34.4	14.3	30.5
		MC-CNN+RBS [41]	**19.5**	9.76	26.1	32.6	**12.2**	**12.4**	**17.9**	**22.7**	19.8	**25.1**	**22.8**	20.4	12.9	**32.4**	14.4	**27.3**
		MeshStereo [42]	21.1	10.1	20.1	36.6	14.1	14.6	19.9	28.9	24.0	32.0	26.7	16.6	16.5	41.0	13.9	27.5
		TMAP [36]	22.9	14.1	21.2	34.0	14.8	14.4	21.0	32.0	20.1	31.2	40.4	21.2	16.1	46.6	14.3	40.3
	Loc.	IPMS (proposed)	20.9	**9.65**	**15.46**	**25.81**	15.93	21.16	19.01	47.18	23.89	33.99	26.73	**13.62**	14.25	50.18	**8.48**	32.36
		IDR [43]	23.0	14.5	21.4	33.3	14.8	12.9	20.8	29.9	23.1	31.1	53.0	16.6	15.9	49.7	13.2	38.4
Average error	Global	MC-CNN-acrt [40]	11.8	4.24	18.7	34.1	7.21	7.22	6.00	9.35	13.5	18.3	9.71	9.37	4.64	6.62	9.35	21.6
		MC-CNN+RBS [41]	6.67	2.22	8.42	22.2	3.95	**3.87**	**2.34**	**4.74**	**13.9**	9.76	4.80	3.66	2.38	**4.63**	**5.90**	**5.13**
		MeshStereo [42]	7.59	2.39	6.44	36.4	5.40	5.71	3.25	5.45	11.6	6.34	4.92	2.73	2.25	11.1	1.90	5.62
		TMAP [36]	7.88	2.44	5.88	30.9	4.72	4.41	3.86	13.7	9.25	**6.12**	16.6	3.13	2.22	10.9	2.73	10.5
	Loc.	IPMS (proposed)	**5.79**	**1.79**	**4.16**	**17.9**	**3.56**	4.68	3.87	13.95	**8.42**	10.19	**3.11**	**2.00**	**2.42**	**8.4**	**1.35**	7.75
		IDR [43]	8.57	2.60	5.97	30.0	4.26	3.90	4.39	10.8	10.4	6.3	39.6	2.61	2.42	10.2	2.54	9.09

There is a large variability in the error bars because of the different individual situations at each sample. We also found out that BOBYQA does not increase the computational cost compared to the Luus-Jaakola method, as it also takes on average 9 iterations to converge.

Fig. 11 shows that IPMS converges in only two iterations on the same subset of images. The main reason behind this faster convergence is the constrained initialization of plane parameters in contrast to the random initialization used in PMS.

Our unoptimised C implementation of PMS takes around 150 minutes to process an image of size 900×750 pixels whereas IPMS is approximately five times more expensive but still comparable to global methods such as [36]. The main reason for the additional computational time is the inclusion of the search image support weight in the cost function, which reduces false matching in occluded regions. Comparing IPMS to IBS clearly shows that the results are improved by adding the support image weights, especially for the `Jadeplant` stereo pair whose background is less textured with big occlusion. Importantly, even if we do not include the support weight for the search image in the cost function, our disparity map produces $15-35\%$ fewer bad pixels in one-third less time than PMS (Fig. 9).

C. Comparison with other methods

We now compare IPMS against state-of-the-art algorithms according to the Middlebury benchmark. Fig. 12 shows all the propagation results per iteration for the Teddy stereo pair reference image (900×750). Our method produces a very good estimate of the disparity map even in the first iteration, because of the constrained initialization.

To further illustrate the performance of our proposed method, we present the disparity map, the error map and the plane distribution map on a subset of the Middlebury training images in Fig. 13. Note that the proposed method successfully tackles the edge fattening problem, *e.g.,* `PlaytableP`. However, as with other local methods our algorithm has difficulties with significant radiometric changes and low textured regions, *e.g.,* `Vintage`. The percentage of bad pixels and average error on all pixels (without using a mask) for the Middlebury training dataset are shown in Table IV along with other top performing published local and global methods.

Results show that our method outperforms other local methods and our results are among the top five considering also global methods.

VII. CONCLUSION

We have proposed a constrained initialization scheme for the plane parameters in PatchMatch stereo so that only feasible planes are associated with each pixel. We used the gradient free non-linear optimiser BOBYQA, which we have shown to be more effective than Luus-Jaakola in refining the plane parameters. Moreover, to tackle imperfectly rectified image pairs, we relaxed the view propagation. These modifications help our method to generate better disparity maps than state-of-the-art local methods, and to converge in only two iterations.

As future work we will extend our method to multi-view stereo and deal with disparity changes in moving scenes. Other possible improvements include an optimised implementation on a GPU architecture, to reduce the computation time.

ACKNOWLEDGEMENTS

Miles Hansard was partially supported by EPSRC grant EP/M01469X/1.

REFERENCES

[1] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon, "KinectFusion: Real-time dense surface mapping and tracking," in *Proc. ISMAR*, Oct 2011, pp. 127–136.

[2] C. Castejón, B. L. Boada, D. Blanco, and L. Moreno, "Traversable region modeling for outdoor navigation," *Journal of Intelligent and Robotic Systems*, vol. 43, no. 2, pp. 175–216, 2005.

[3] A. Segal, D. Hhnel, and S. Thrun, "Generalized-ICP." in *Robotics: Science and Systems*. The MIT Press, 2009.

[4] H. Sekkati and S. Negahdaripour, *3-D Motion Estimation for Positioning from 2-D Acoustic Video Imagery*. Springer Berlin Heidelberg, 2007.

[5] R. Hoffman and A. K. Jain, "Segmentation and classification of range images," *IEEE Trans. PAMI*, vol. 9, no. 5, pp. 608–620, May 1987.

[6] M. Hebert, "Outdoor scene analysis using range data," in *Proc. ICRA*, vol. 3, Apr 1986, pp. 1426–1432.

[7] M. Bleyer and C. Breiteneder, "Stereo matching-state-of-the-art and research challenges," in *Advanced Topics in Computer Vision*. Springer London, 2013, pp. 143–179.

[8] M. Bleyer, C. Rhemann, and C. Rother, "PatchMatch stereo - stereo matching with slanted support windows," in *Proc. BMVC*, 2011.

[9] P. Heise, S. Klose, B. Jensen, and A. Knoll, "PM-Huber: PatchMatch with Huber regularization for stereo matching," in *Proc. ICCV*, Dec 2013, pp. 2360–2367.

[10] F. Besse, C. Rother, A. Fitzgibbon, and J. Kautz, "PMBP: PatchMatch belief propagation for correspondence field estimation," *Int. J. Comput Vision*, vol. 110, no. 1, pp. 2–13, 2014.

[11] S. Galliani, K. Lasinger, and K. Schindler, "Massively parallel multiview stereopsis by surface normal diffusion," in *Proc. ICCV*, 2015, pp. 873–881.

[12] D. Scharstein and R. Szeliski, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," *Int. J. Comput Vision*, vol. 47, no. 1-3, pp. 7–42, 2002.

[13] R. Luus and T. H. I. Jaakola, "Optimization by direct search and systematic reduction of the size of search region," *AIChE Journal*, vol. 19, no. 4, pp. 760–766, 1973.
[14] Y. HaCohen, E. Shechtman, D. B. Goldman, and D. Lischinski, "Non-rigid dense correspondence with applications for image enhancement," *ACM Trans. Graph.*, vol. 30, no. 4, Jul. 2011.
[15] M. J. D. Powell, "The BOBYQA algorithm for bound constrained optimization without derivatives," *(Report No. DAMTP 2009/NA06). Centre for Mathematical Sciences, University of Cambridge, UK.*, Aug 2009.
[16] Y. Yuan, "Recent advances in trust region algorithms," *Mathematical Programming*, vol. 151, no. 1, pp. 249–281, 2015.
[17] L. Nalpantidis, C. S. Georgios, and A. G., "Review of stereo vision algorithms: From software to hardware," *Int. J. Optomechatronics*, vol. 2, no. 4, pp. 435–462, 2008.
[18] A. Klaus, M. Sormann, and K. Karner, "Segment-based stereo matching using belief propagation and a self-adapting dissimilarity measure," in *Proc. ICPR*, vol. 3, 2006, pp. 15–18.
[19] H. Hirschmuller, "Stereo processing by semiglobal matching and mutual information," *IEEE Trans. PAMI*, vol. 30, no. 2, pp. 328–341, Feb 2008.
[20] C. Rhemann, A. Hosni, M. Bleyer, C. Rother, and M. Gelautz, "Fast cost-volume filtering for visual correspondence and beyond," in *Proc. CVPR*, June 2011, pp. 3017–3024.
[21] D. Scharstein, R. Szeliski, and H. Hirschmuller, *Middlebury Stereo Vision*, [Last accessed Dec.-2017] http://vision.middlebury.edu/stereo/.
[22] S. Yoon, S. Park, S. Kang, and Y. Kwak, "Fast correlation-based stereo matching with the reduction of systematic errors," *Pattern Recognition Letters*, vol. 26, no. 14, pp. 2221–2231, 2005.
[23] K. Mühlmann, D. Maier, J. Hesser, and R. Männer, "Calculating dense disparity maps from color stereo images, an efficient implementation," *Int. J. Comput. Vision*, vol. 47, no. 1-3, pp. 79–88, Apr. 2002.
[24] P. Mordohai and G. Medioni, "Stereo using monocular cues within the tensor voting framework," *IEEE Trans. PAMI*, vol. 28, no. 6, pp. 968–982, June 2006.
[25] K. Yoon and I. Kweon, "Adaptive support-weight approach for correspondence search," *IEEE Trans. PAMI*, vol. 28, no. 4, 2006.
[26] A. Hosni, M. Bleyer, C. Rhemann, M. Gelautz, and C. Rother, "Real-time local stereo matching using guided image filtering," in *Proc. ICME*, July 2011, pp. 1–6.
[27] D. Gallup, J. M. Frahm, P. Mordohai, Y. Qingxiong, and M. Pollefeys, "Real-time plane-sweeping stereo with multiple sweeping directions," in *Proc. CVPR*, June 2007, pp. 1–8.
[28] C. Barnes, E. Shechtman, A. Finkelstein, and D. Goldman, "PatchMatch: A randomized correspondence algorithm for structural image editing," *ACM Trans. Graph.*, vol. 28, no. 3, p. 24, 2009.
[29] G. Li and S. Zucker, "Differential geometric inference in surface stereo," *IEEE Trans. PAMI*, vol. 32, no. 1, pp. 72–86, Jan 2010.
[30] Y. Huang, M. K. Ng, and Y. W. Wen, "A fast total variation minimization method for image restoration," *Multiscale Modeling & Simulation*, vol. 7, no. 2, pp. 774–795, 2008.
[31] H. Hirschmuller and D. Scharstein, "Evaluation of stereo matching costs on images with radiometric differences," *IEEE Trans. PAMI*, vol. 31, no. 9, pp. 1582–1599, Sept 2009.
[32] A. Hosni, M. Bleyer, and M. Gelautz, "Secrets of adaptive support weight techniques for local stereo matching," *Comput. Vis. Image Underst.*, vol. 117, no. 6, pp. 620 – 632, 2013.
[33] M. J. Black and P. Anandan, "The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields," *Comput. Vis. Image Underst.*, vol. 63, no. 1, pp. 75–104, 1996.
[34] M. J. Black, "Robust incremental optical flow," Ph.D. dissertation, Yale University, New Haven, CT, USA, 1992.
[35] T. Kanade and M. Okutomi, "A stereo matching algorithm with an adaptive window: Theory and experiment," *IEEE Trans. PAMI*, vol. 16, no. 9, pp. 920–932, Sep. 1994.
[36] E. T. Psota, J. Kowalczuk, M. Mittek, and L. C. Perez, "MAP disparity estimation using hidden markov trees," in *Proc. ICCV*, 2015, pp. 2219–2227.
[37] C. Stewart, R. Flatland, and K. Bubna, "Geometric constraints and stereo disparity computation," *Int. J. Comput Vision*, vol. 20, no. 3, pp. 143–168, 1996.
[38] G. Marsaglia, "Choosing a point from the surface of a sphere," *Ann. Math. Statist.*, vol. 43, no. 2, pp. 645–646, 1972.
[39] S. Johnson, *The NLopt nonlinear-optimization package*, http://ab-initio.mit.edu/nlopt.
[40] J. Zbontar and Y. LeCun, "Stereo matching by training a convolutional neural network to compare image patches," *CoRR*, vol. abs/1510.05970, 2015, http://arxiv.org/abs/1510.05970.
[41] J. Barron and B. Poole, "The fast bilateral solver," in *Proc. ECCV*, 2016.
[42] C. Zhang, Z. Li, Y. Cheng, R. Cai, H. Chao, and Y. Rui, "MeshStereo: A global stereo model with mesh alignment regularization for view interpolation," in *Proc. ICCV*, June 2015.
[43] J. Kowalczuk, E. T. Psota, and L. C. Perez, "Real-time stereo matching on CUDA using an iterative refinement method for adaptive support weight correspondences," *IEEE Trans. CSVT*, vol. 23, no. 1, pp. 94–104, 2013.