

Efficient Dense-Stereo and Novel-view Synthesis for Gaze
Manipulation in One-to-one Teleconferencing

A. Criminisi, J. Shotton, A. Blake, C. Rother, P.H.S. Torr

`antcrim@microsoft.com`

Sep 2003

Technical Report

MSR-TR-2003-59

A new algorithm is proposed for novel-view synthesis, with particular application to teleconferencing. Given the video streams acquired by two cameras placed on either side of a computer monitor, the proposed algorithm synthesises images from a virtual camera in arbitrary position (typically located within the monitor area) to facilitate eye contact. The new technique is based on an improved, dynamic-programming, stereo algorithm for efficient novel-view generation. The two main contributions of this paper are: i) a new four-layer matching graph for dense-stereo dynamic-programming, that supports accurate occlusion labeling; ii) a compact geometric derivation for novel-view synthesis by *direct* projection of the *minimum-cost surface*. Furthermore, the paper presents an algorithm for the temporal maintenance of a background model to enhance the rendering of occlusions and reduce temporal artefacts (flicker); and a cost aggregation algorithm that acts directly in three-dimensional matching cost space. The proposed algorithm has been designed to work with input images with large disparity range, a common situation in one-to-one video-conferencing. The enhanced occlusion-handling capabilities of the new DP algorithm are evaluated against those of the most powerful state-of-the-art dynamic-programming and graph-cut techniques. A number of examples demonstrate the robustness of the algorithm to artefacts in stereo video streams. This includes demonstrations of cyclopean view synthesis in extended conversational sequences, synthesis from a freely translating virtual camera and, finally, basic 3D scene editing.

Microsoft Research Ltd.
7 J J Thomson Ave
Cambridge, UK CB3 0FB
<http://www.research.microsoft.com>

Contents

1	Introduction	1
2	Background on DP and Novel-view Synthesis	4
2.1	Conventional dynamic-programming	4
2.2	Direct cyclopean-view synthesis from DP	7
3	Occlusion classification via the <i>four-move</i> model	9
4	Imposing constraints on occlusions by DP on a <i>four-plane</i> graph	10
5	Matching cost aggregation	14
5.1	Inter-scanline consistency and cost aggregation	16
6	Evaluating recovered occlusion maps	17
6.1	The advantages of the 4-move, 4-plane model	17
6.2	Estimating the occlusion map of a standard test stereo pair.	19
6.3	Comparisons with state of the art.	19
7	Rendering occlusions	23
8	Rendering from variable viewpoint	27
9	Results: novel-view synthesis	30
10	Conclusions and future work	34

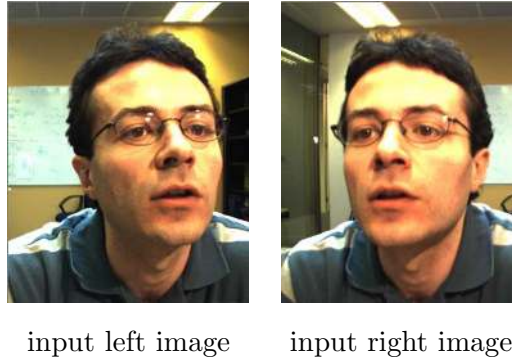


Figure 1: Lack of eye contact. *In one-to-one video-conferencing, cameras located on the frame of the computer monitor fail to capture gaze correctly. Here a person looks at the centre of the screen but, in the images captured by cameras mounted on either side of the computer monitor, he does not appear to be looking directly ahead. The aim of the proposed algorithm is to synthesize a virtual view that procures eye contact.*

1 Introduction

This paper addresses the problem of novel-view synthesis from a pair of rectified images with specific emphasis on gaze correction for one-to-one teleconferencing. With the rise of instant messaging technologies¹, it is envisaged that the PC will increasingly be used for interactive visual communication. One pressing problem is that any camera used to capture images of one of the participants has to be positioned offset from his or her gaze (*cf.* fig. 1 and fig. 2). This can lead to lack of eye contact and hence undesirable effects on the interaction [7].

One might think that if it were possible to drill a hole in the centre of a computer screen and place a camera there, that would achieve the desired viewpoint. The first problem with this solution is that “porous” screens do not exist; but even if they did the user would be required always to look at the centre of the monitor, where the extra camera had been inserted. However in a messaging session, the user looks at the communication window (where the other person’s face appears) which can be displaced and moved around the screen at will (fig. 2a). Therefore, the camera needs to be placed behind the messaging window on the screen but this cannot be achieved with available hardware and therefore a software solution is sought.

Previously proposed approaches can be broadly categorized as *model-based* or *image-based*. One model-based technique is to use a detailed 3D head model, texture map it and

¹*e.g.* messenger.msn.co.uk/, messenger.yahoo.com/, www.aol.co.uk/aim/

reproject it into the required viewpoints. Whilst this can be successful [18, 19], it is limited to imaging heads with no hair or neck. Nor can it deal with occlusion events such as a hand in front of the face. A more general approach, proposed here, is to use image-based rendering techniques (*IBR* [3]) to synthesize novel views from two input images. The entire input images, as opposed to the head only, are processed, thus avoiding the detection and modeling of heads with all the associated problems.

In IBR a depth map is combined with input images to produce synthetic views. In order to generate a depth map a dense stereo algorithm is required, a substantial review of which can be found in [15], in which the authors categorize and evaluate a number of existing dense-stereo techniques. But this evaluation may not be sufficient for our purposes as: (i) the range of disparities considered in [15] is much smaller than in our application (0-29 pixels there, whereas we typically consider 0-80 pixel disparities); (ii) we are primarily interested in new-view synthesis so it does not matter if the disparities are relatively inaccurate in texture-less image regions; all that matters is that the new view is well synthesized (as noted in [14, 17]); (iii) we consider long video sequences so temporal stability also plays an important part.

In the past, research on dense stereo reconstruction has been directed largely towards the accurate recovery of disparity maps, though not entirely [1]. We have found that while inaccurate disparities may still produce acceptable synthesized images over matched regions, inconsistent occlusion maps lead to unacceptable artefacts.

According to the evaluation in [15], two of the most powerful dense stereo techniques use Graph cut (GC) [11, 13] and loopy belief propagation [16]. Both of these are currently too computationally intensive for real-time applications and one of the goals of this paper is to produce high-quality synthetic sequences at frame rate.

One of the most computationally efficient algorithms for stereo is Epipolar-line Dynamic Programming [12], commonly referred to as DP, and that is the basis of the work reported here. The DP algorithm described in [5] has previously been demonstrated for cyclopean view interpolation [4] in video². In the basic form of the DP algorithm, in order to obtain computational efficiency observations consist of single-pixel intensities. This, together with the fact that pairs of corresponding scanlines are considered independently introduces a number of artefacts which corrupt the quality of the output reconstruction (especially for large disparity ranges) as fig. 3b shows.

²We refer to *cyclopean view* as the image generated from a virtual camera located in the mid-point between the two input cameras.

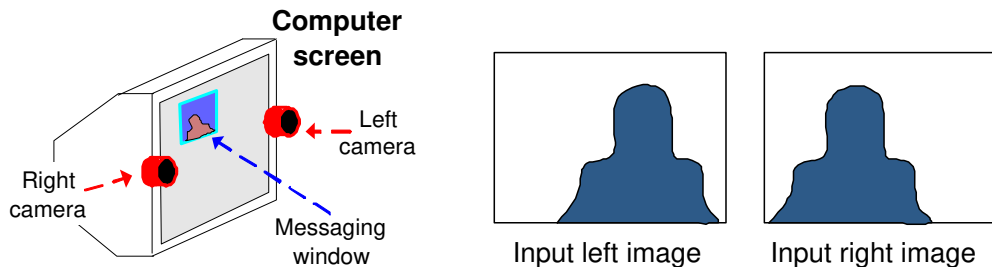


Figure 2: Camera-computer configuration. *The basic setup considers two cameras placed on the frame of the computer monitor. A window for the one-to-one teleconferencing application is marked in blue on the computer screen. The goal of this paper is that of facilitating eye contact by efficient processing of the two input images to generate high-quality views for virtual cameras placed behind the messaging application window. The technique described in this paper achieves gaze correction in an efficient and compelling way.*

In particular, DP-based algorithms for novel-view synthesis are characterized by three kinds of artefacts: (i) artefacts produced by mismatches (horizontal streaks due to inconsistencies between adjacent scanlines); (ii) the “halo” in the regions where the background is visible in only one of the two input views (occlusions); and (iii) flickery synthetic pixels, caused by matching ambiguities. The first two kinds of artefacts are static, while the latter is temporal in that it appears when processing sequences of stereo images. This paper sets out to address and solve all three kinds of artefacts while maintaining high computational efficiency.

We present new contributions in two areas: accurate generation of disparity and occlusion maps and efficient new-view rendering. For the first we propose a new DP algorithm acting on a *four*-plane graph (as opposed to the conventional single-plane DP). New labels are introduced for occlusions, and the cost function is extended to favour: (a) good grouping of occlusions, (b) formation of solid occlusion regions at the boundaries of foreground objects, and (c) inter-scanline consistency. For the second aspect we introduce *minimum-cost surface projection* as a compact technique for generating synthetic views from arbitrary virtual cameras, *directly* from the minimum-cost surface obtained during the DP process³. This technique avoids the explicit construction of a 3D mesh model or depth map.

³The *minimum-cost surface* is defined to be the collection of all the minimum-cost paths estimated (independently) by the DP algorithm at each scanline.

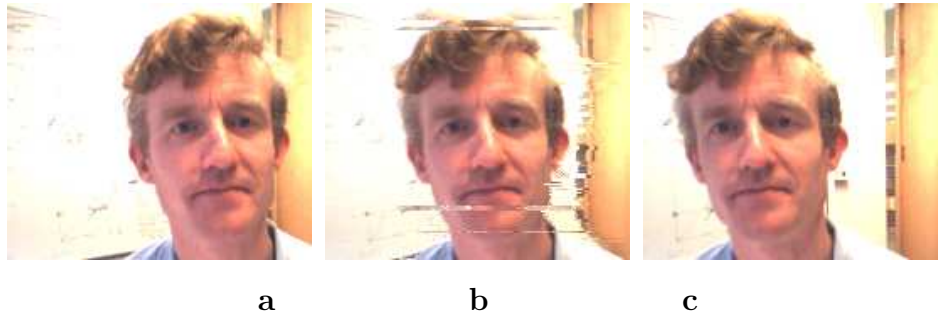


Figure 3: Fast cyclopean view synthesis by dynamic-programming. *(a,c) Input left and right views, respectively; (b) Cyclopan view synthesized by standard dynamic-programming [4]. Note that gaze is correct in the cyclopean view. The algorithm runs at near real-time rate, but produces considerable artefacts (horizontal streaks) in the synthesized cyclopean image.*

Paper outline Section 2 reviews the state of the art in dense stereo via dynamic-programming, and consequent issues for novel view rendering, particularly of occluded objects. The main contribution of this paper is described in sections 3 and 4 which introduce our improved multi-layer, dense-stereo algorithm. Section 5 illustrates the cost filtering algorithm for inter-scanline consistency. Section 6 presents an evaluation of performance of our technique with respect to occlusion detection, together with a comparison with state of the art dynamic-programming and graph-cut (GC) algorithms. Our technique for synthesising occluded regions is described in section 7 and the virtual-view generation and rendering algorithm is described in section 8. Finally, section 9 demonstrates the effectiveness of the proposed techniques with a number of examples where both static images and entire sequences are generated for various virtual camera locations.

2 Background on DP and Novel-view Synthesis

This section reviews the principles of dynamic-programming (DP) algorithms for dense stereo [5,12] and discusses issues related to the synthesis of virtual cyclopean views from the two input images.

2.1 Conventional dynamic-programming

Figure 4 shows a plan view of the camera setup. The left and right cameras provide us with the synchronized and epipolar-rectified [8] input videos. The focal length is denoted f , and B

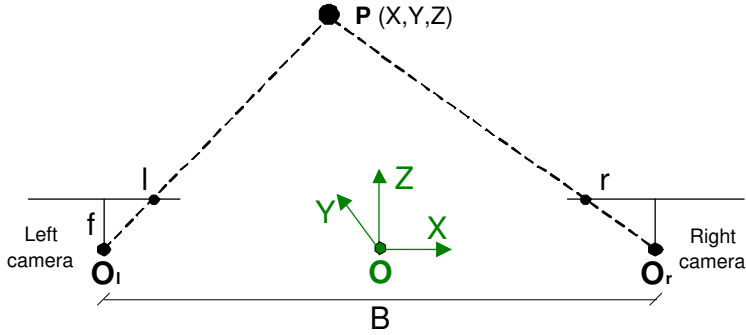


Figure 4: Basic camera configuration and notation. O_l and O_r are the optical centres of left and right cameras respectively, f is the focal length of the cameras (identical for both cameras) and B is the baseline between the two optical centres. The origin of the reference coordinate system X, Y, Z is denoted O .

is the distance between the two optical centres (the baseline). A Cartesian coordinate system is chosen with origin at the mid-point between the left and right optical centres. A 3D scene point P is projected into the two input image planes in corresponding image points at l and r distances from the respective image centres. The distance $d = l - r$ is commonly known as disparity. We refer to the images corresponding to a virtual camera, with optical centre in the origin O , as *cyclopean* images. As will be demonstrated, our algorithm is not restricted to cyclopean views only but is capable of generating virtual images from arbitrary viewpoints. The diagram in fig. 5a represents the matching graph for a pair of corresponding scanlines in the two input images [5, 12]. Note that, since $l \geq r \quad \forall P$ (i.e. disparities $d = l - r$ are always non-negative), then it is only ever necessary to consider the lower half of the matching graph (grey area in fig. 5a). The limiting, zero-disparity case $l = r$ corresponds to points at infinity. The 45-degree line in fig. 5a is termed the “virtual scanline” for reasons that will become obvious in the next section. The local cost of matching a pixel at position l along the left scanline with a pixel at position r along the right scanline is denoted $M(l, r)$. In conventional DP, the cost $M(l, r)$ may be defined simply as the square difference of pixel intensities, though more elaborate measures based on patches, colour, wavelets etc., can be used. The cost function employed here is described in section 5.

Standard 3-move DP Dynamic-programming consists of two passes: forward and backward [5]. The forward step constructs a matrix of cumulative matching costs C by the

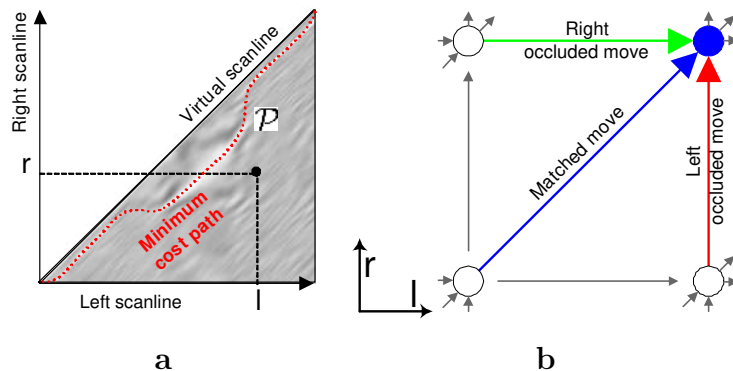


Figure 5: Diagrams for conventional dynamic-programming. (a) The two-dimensional graph on which DP is based. Each node in the planar graph corresponds to a pair of pixels, in the left and in the right scanlines. A matching cost $M(l, r)$ is associated to each node and the goal of DP is finding the minimum-cost path (shown in red) joining the two opposite corners of the graph. Bright pixels correspond to high pixels similarity, i.e. low values of $M(l, r)$. (b) A blown-up view of (a) showing the set of the three allowed moves between pixel pairs [4]. The circles represent nodes of the graph in (a).

following recurrence:

$$C(l, r) = \min \begin{cases} C(l-1, r) & + \beta \\ C(l-1, r-1) & + M(l, r) \\ C(l, r-1) & + \beta \end{cases} \quad (1)$$

where $C(l, r)$ indicates the cumulative cost of the path from the point $(0, 0)$ to the point (l, r) . Notice that only three moves are permitted: a horizontal, possibly *occluded*, move, a diagonal *matched* move and a vertical, possibly *occluded* move (fig. 5b). Thus, 45-degree segments in the minimum cost path correspond to fronto-parallel surfaces (constant disparity); vertical and horizontal segments represent either occlusions or non-fronto-parallel surfaces. The cost of a horizontal/vertical move, which may indicate occlusion, is β , a manually set parameter. When matching costs $M(l, r)$ are normalised so that $0 \leq M(l, r) \leq 1$, a value of $\beta = 0.3$ seems to yield the best results on a variety of images. At each iteration the minimum cost between the three possible moves is chosen and a table of backwards links is stored for use in the second part of DP.

The backward pass of the algorithm follows the saved back-links from $(l = W, r = W)$ to the origin $(l = 0, r = 0)$ and defines the minimum-cost path \mathcal{P} as the sequence of visited nodes. Knowing \mathcal{P} is equivalent to knowing dense correspondences between pixels in the

For each pair of scanlines, given their matching path \mathcal{P} :

- For each point $\mathbf{p} \in \mathcal{P}$
 1. take the colours $I^l(l)$ and $I^r(r)$ of the corresponding pixels l and r in the left and right scanlines, respectively;
 2. compute the average value $\tilde{I} = \frac{1}{2}(I^l(l) + I^r(r))$;
 3. project the newly obtained pixel orthogonally to the virtual image scanline, into the virtual image point \mathbf{v} ; *i.e.* $I^v(v) = \tilde{I}$.

Table 1: Cyclopean view synthesis from direct projection of the minimum-cost path.

input images and thereby the disparity map.

Limitations of conventional DP The three-move model is limited since it fails to distinguish completely between occluded and non-occluded moves. One of the main contributions of this paper will be to expand the set of permitted moves to support unambiguous detection and classification of occlusion events.

2.2 Direct cyclopean-view synthesis from DP

This introductory section explains how cyclopean views can be generated directly from the minimum-cost paths estimated by conventional DP. Special attention is paid to the synthesis of pixels in occluded regions. The basic cyclopean-view synthesis algorithm is described in Table 1 and illustrated in fig. 6a.

The algorithm in Table 1 applies to matched pixels *only* and occluded areas must be treated differently. Indiscriminate application of the algorithm in Table 1 to occluded and unoccluded points alike, would produce a distorting effect, a “halo” around the foreground objects in the cyclopean view. An example is shown in fig. 7 where the frame of the door and the edge of the whiteboard have been deformed into curves which follow the outline of the foreground head. The “halo” artefact is much more noticeable and disturbing when video sequences are reconstructed in this way.

Fronto-parallel assumption for occlusion filling In order to overcome the halo effect occluded pixels must first be detected. For those pixels it is necessary to make a plausible assumption about underlying 3D structure since this information is not available given

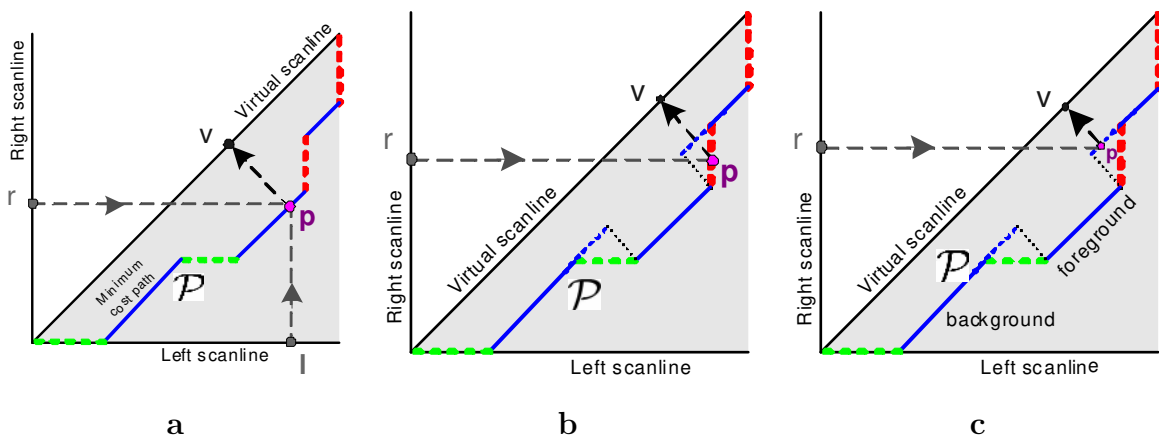


Figure 6: Generating the cyclopean view. (a) A matched point $\mathbf{p} \in \mathcal{P}$ is projected orthogonally onto its corresponding point v on the virtual scanline. The luminance value of the virtual pixel v is the average of the corresponding pixels l and r on left and right images, respectively. (b) Halo: treating the occluded segments in \mathcal{P} in the same way as the matched segments produces a lens-like effect that we call the “halo” artefact (fig. 7). (c) Fronto-parallel occlusion synthesis: the halo effect is largely removed if a fronto-parallel background assumption is made: an occluded point \mathbf{p} on the continuation of the background is projected orthogonally onto its corresponding point v on the virtual scanline.

the absence of a stereo match. One effective assumption is that of a fronto-parallel background [14]. As illustrated in fig. 6c, filling of the occluded regions can be achieved under the fronto-parallel assumption by extending the background at constant disparity. Fig. 6c shows how, for a left occlusion (vertical dashed line), the values of the virtual pixels are taken *only* from the right image: $I^v(v) = I^r(r)$, and vice-versa.

The reset artefact The fronto-parallel approximation can be applied only if occluded regions are correctly detected. Detection errors (fig. 8b) cause the sampling of “source” pixel values from incorrect locations in the input images — the *reset* artefact. Accurate detection of occlusion is clearly paramount.

The next two sections introduce our improved DP algorithm for accurate occlusion detection.

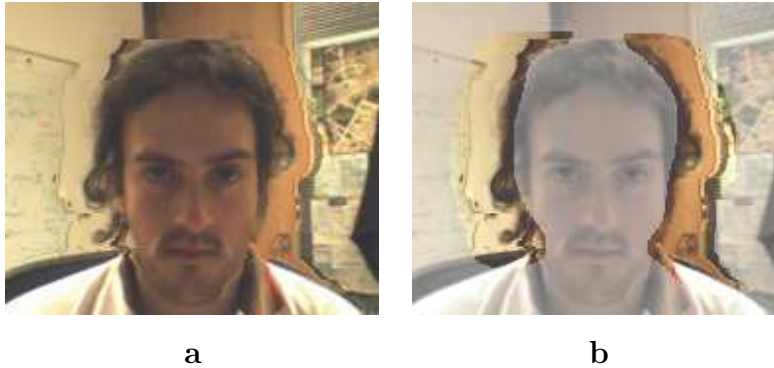


Figure 7: Halo artefact. (a) A cyclopean view reconstructed by applying the algorithm in Table 1 to both matched and occluded segments of the recovered minimum-cost path. A “halo” of deformed background objects is visible around the head. (b) Regions over which the halo effect occurs are highlighted.

3 Occlusion classification via the *four-move* model

A major drawback to the standard DP approach is that slanted surfaces in space are matched as a combination of diagonal and horizontal or vertical moves in the matching graph. So, as shown in fig. 9b, horizontal/vertical moves do not necessarily indicate occlusion. In order to disambiguate between horizontal/vertical *matched* moves and true occlusion events we augment the basic three-move model. The improved model has two types of horizontal moves and two types of vertical moves (matched and occluded). Since a line at any orientation can always be approximated by a sequence of horizontal and vertical matched moves (fig. 9c), the diagonal matched move of the basic DP model is eliminated without loss. This defines the *four-label model* of fig. 10, to be compared with fig. 5b.

In recent work [6] we have tried the five-move model (including a matched diagonal move) as suggested also by Ishikawa *et al.* [9]; but we have found the four-move model as reliable as the five-move model and simpler. In the four-move model, every possible path through the cost space has equal length (Manhattan distance) so that the costs of alternative paths are truly comparable.

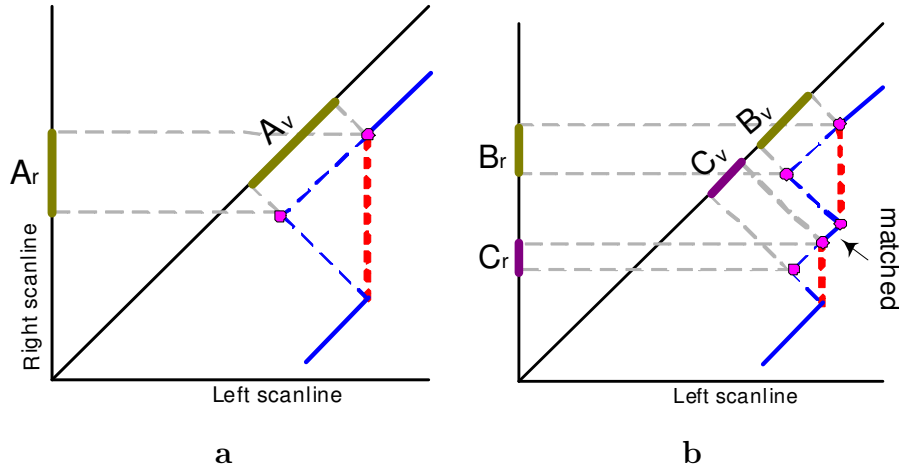


Figure 8: The reset effect. (a) The fronto-parallel approximation used for filling occluded regions. For the left occlusion marked in red (dashed), the region A_v in the cyclopean image is copied from the corresponding region A_r in the right image. See also fig. 6c. (b) A small error in the detection of the occluded region, e.g. a small matched region inside an occluded region, produces a large error in the cyclopean reconstruction. In fact, the “source” regions B_r and C_r are quite different from A_r and far apart from each other. This produces visible artefacts as illustrated later in the paper.

4 Imposing constraints on occlusions by DP on a *four-plane* graph

Matches and occlusions are now unambiguously labelled here, unlike the three-move model. The three-move DP algorithm conventionally runs on a single planar graph. Four move DP is best visualised in a four-plane graph with two occluded planes L_0 (occluded in left image) and R_0 (occluded in right image) and two matched planes L_m and R_m . The optimal path through the matching graph evolves in this three-dimensional space (see fig. 11 and fig. 13).

The four-plane model reflects naturally the persistence of each of the states. For instance long runs of occlusions can be favoured by setting a high cost for entering or leaving an occluded state (L_o or R_o). Similarly, it is desirable to bias *against* runs of matched moves in R_m or L_m , ensuring that surfaces close to fronto-parallel are preferred, as in the original three-move DP. Slanted surfaces are thus described by oscillations of the optimal path between L_m and R_m . The 4-move framework includes four different cumulative cost matrices: C_{L_o} , C_{R_o} , C_{L_m} and C_{R_m} , one for each plane in the graph. The elements of the cumulative cost matrices

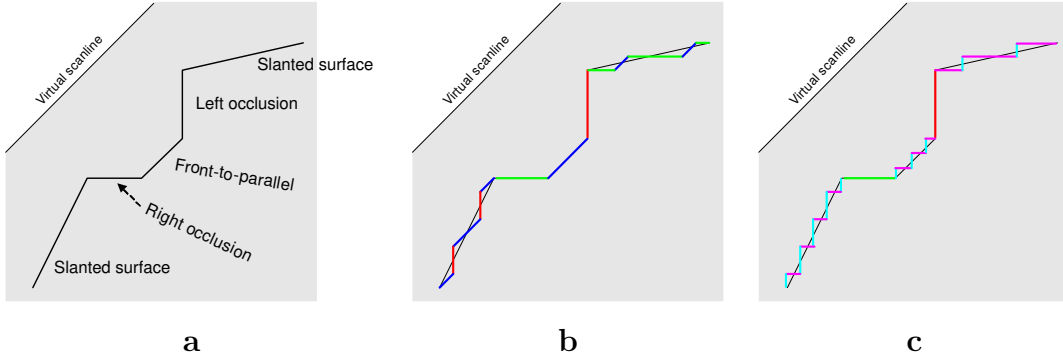


Figure 9: (a) Vertical and horizontal segments in the matching path correspond to occlusions. Diagonal 45-degree segments to fronto-parallel surfaces and other orientations to slanted surfaces. (b) The conventional 3-move DP algorithm approximates the matching path with a combination of the three basic moves. Slanted surfaces must be approximated by matched and occluded moves, where the occluded moves do not correspond to genuine occlusions (where a point is seen by only one camera). (c) Our improved four-move models tends to label occlusions correctly. Slanted surfaces are approximated by a combination of matched moves only.

are initialised to $+\infty$ everywhere except in one row of the right occluded plane, where:

$$\mathbf{C}_{Ro}[i, 0] = i\alpha \quad \forall i = 0 \dots W - 1 . \quad (2)$$

The forward step of our DP algorithm computes the four cumulative cost matrices according to the following recursion:

$$\mathbf{C}_{Lo}[l, r] = \min \begin{cases} \mathbf{C}_{Lo}[l, r - 1] + \alpha \\ \mathbf{C}_{Lm}[l, r - 1] + \beta \\ \mathbf{C}_{Rm}[l, r - 1] + \beta \end{cases} \quad (3)$$

$$\mathbf{C}_{Lm}[l, r] = M(l, r) + \min \begin{cases} \mathbf{C}_{Lo}[l, r - 1] + \beta' \\ \mathbf{C}_{Lm}[l, r - 1] + \gamma \\ \mathbf{C}_{Rm}[l, r - 1] \\ \mathbf{C}_{Ro}[l, r - 1] + \beta' \end{cases}$$

where $M(l, r)$ is the cost of matching the l^{th} pixel in the left scanline with the r^{th} pixel in the right scanline. In this section we are assuming given the matching costs $M(l, r)$ and focus on the DP algorithm only. Section 5 will describe how the cost function is computed.

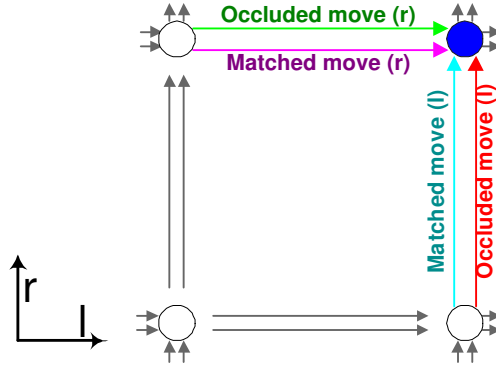


Figure 10: The four-move model for DP allows two matched moves (marked in magenta and cyan), and two occluded moves (green and red).

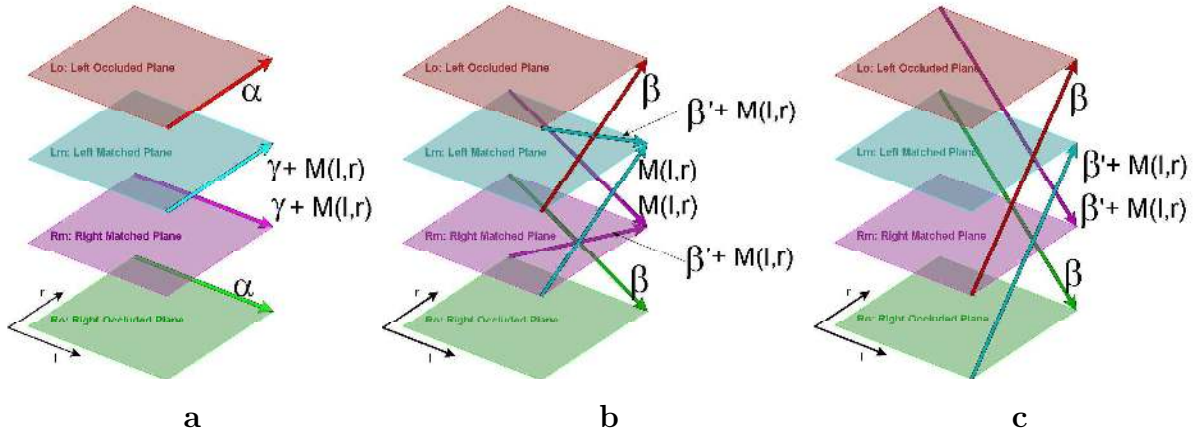


Figure 11: The proposed four-plane model for DP. The graph associated with 4-move DP now occupies four planes, with allowed moves (a) within planes and (b,c) between planes. The permitted moves, shown by arrows, have been labelled with the associated costs — see text.

The two other cost matrices $C_{Ro}[l, r]$ and $C_{Rm}[l, r]$ are defined by invoking symmetry on the definitions of $C_{Lo}[l, r]$ and $C_{Lm}[l, r]$ above. The cost structure defined by the four plane DP algorithm can be pictured as a finite state machine as in figure 12.

In this forward pass, the cost matrix computation proceeds from the corner $(l = 0, r = 0)$ in the left occluded plane (L_o) and continues up to $(l = W - 1, r = W - 1)$ in the right occluded plane (R_o), where W is the image width. As cumulative costs matrices are built, the algorithm stores, at each iteration, backward link pointers to the node with minimum cumulative cost, as in conventional DP. In the backward pass, the minimum-cost path is recovered by following the fourteen different kinds of back-links from the $(l = W - 1, r =$

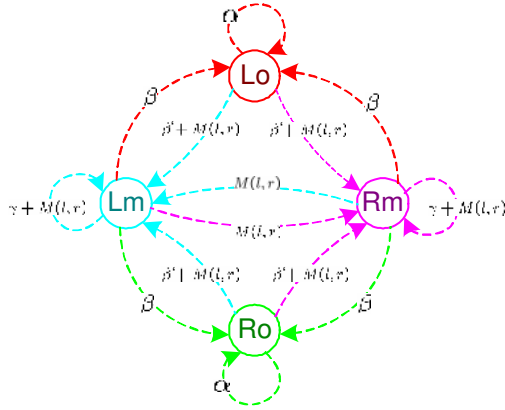


Figure 12: Finite State Machine Representation. Our four-move, four-plane algorithm can be represented as a finite state machine with four states. The four states represent the four permitted moves. The edge labels represent the costs associated to the different transitions between states.

$W - 1$) corner on the left-occluded plane L_o to $(l = 0, r = 0)$ on the right-occluded plane R_o .

Setting the transition costs The penalty parameters α , β , β' and γ are chosen as follows:

- The parameter α is set to $1/2$, a value chosen just sufficient to exceed the typical cost $M(l, r)$ ($0 \leq M(l, r) \leq 1$) of a good match.
- The penalty cost β is set to 1.0 – large enough to avoid erroneous labelling of weak true matches as occlusions, but not so large as to prevent the minimum cost path ever entering an occluded plane.
- The parameter β' is set to 1.0 – large enough to avoid reset artefacts (leaving an occluded state too soon), but not so large as to prevent the minimum cost path ever entering a matched plane.
- The cost γ is set to $1/4$ to bias *against* runs of moves within the same *matched* plane. Clearly we do not want to disallow these moves as these are how we approximate slanted surfaces, but it is envisioned that in most cases, the minimum cost path will oscillate between the left and right matched planes, approximating a roughly fronto-parallel surface in a stair-step fashion (see fig. 13d).

Typically, we set $\beta = \beta'$, thus reducing the number of parameters to three. However, different values of β and β' can favour occlusion events ($\beta < \beta'$) or matching events ($\beta > \beta'$). Optimal values for these parameters may be learnt from input ground-truth data.

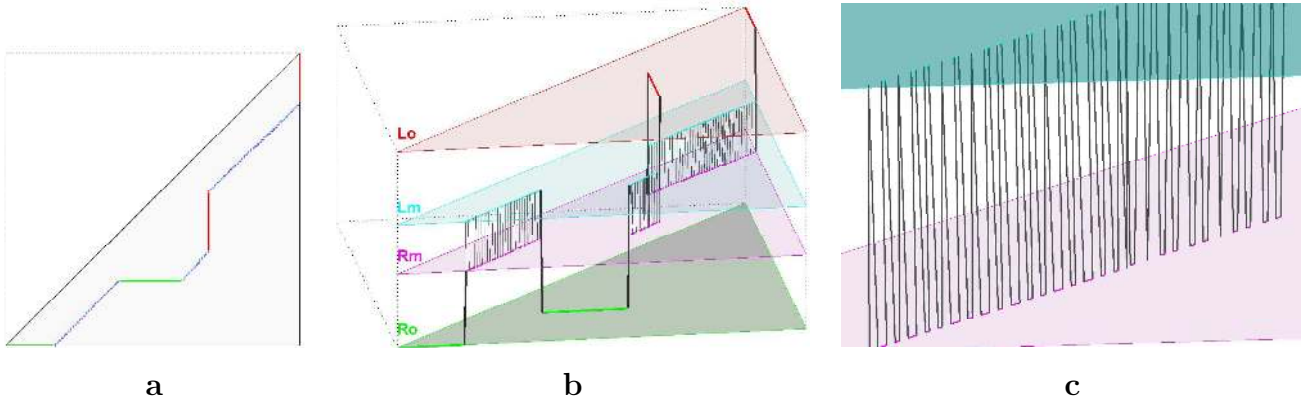


Figure 13: Minimum-cost 3D path in the four-plane graph for DP. (a,b) Different views of the 3D minimum-cost path estimated for a pair of input scanlines. The rapid oscillations between the two matched planes (L_m and R_m) illustrate the way matched, slanted surfaces are represented as alternate horizontal and vertical matched moves. (c) A detail from (b) highlighting the matched oscillations.

Figure 13 shows an example of the recovered minimum-cost 3D path for a pair of corresponding scanlines extracted from real images. The 3D minimum cost path resulting from the application of our DP algorithm weaves its way through the four planes of the graph. Notice the two large occlusions (red and green segments) lying on the corresponding occluded planes. Moreover, as expected, slanted surfaces are tracked as series of oscillations between the two matched planes.

Next, we discuss the details of the cost function construction and cost aggregation.

5 Matching cost aggregation

One of the biggest problems of dynamic-programming, dense stereo algorithms is that scanlines are treated independently. This induces visible “streaky” artefacts in the output disparity maps and related synthesized images. This issue is addressed here by filtering the matching cost matrix across scanlines, over a three-dimensional cost space.

Computation of matching costs The use of neighbourhood windows in computing the cost of matching two pixels has already been shown to help reduce streaky artefacts [15]. The matching cost $M(l, r)$ we employ in this paper is calculated for every pair of pixels along corresponding epipolar lines with a windowed Normalised Sum of Squared Differences

(SSD), defined as:

$$M(l, r) = \frac{M'(l, r)}{2} \quad (4)$$

with

$$M'(l, r) = \frac{\sum_{\epsilon \in \Omega} [(I_{\mathbf{p}_{l+\delta}}^l - \bar{I}_{\mathbf{p}_l}^l) - (I_{\mathbf{p}_{r+\delta}}^r - \bar{I}_{\mathbf{p}_r}^r)]^2}{\sum_{\epsilon \in \Omega} (I_{\mathbf{p}_{l+\delta}}^l - \bar{I}_{\mathbf{p}_l}^l)^2 + \sum_{\epsilon \in \Omega} (I_{\mathbf{p}_{r+\delta}}^r - \bar{I}_{\mathbf{p}_r}^r)^2} \quad (5)$$

where Ω is an $n \times m$ generic template patch centred at the origin of the coordinate system; \mathbf{p}_l and \mathbf{p}_r are the pixels positions (2-vectors) in the left and right images, respectively; and δ is a variable 2D displacement vector. The bar indicates the mean operator.

The mean subtraction and rescaling operations in (6) help deal with changes in the photometric settings of the two input cameras and limited non-Lambertian effects (*e.g.* reflections and specularities). Our experiments show that for horizontally rectified images taller neighborhood windows (*e.g.* 3×7) help incorporate inter-scanline information better than square windows of similar area, with obvious advantages in terms of speed. Furthermore, the costs $M(l, r)$ can be computed efficiently using moving average techniques [15]. The normalization property of (4) ($0 \leq M(l, r) \leq 1 \forall l, r$) will turn out to be extremely convenient when setting the costs of the graph edges defined in the next sections.

In our experiments we have compared the Normalized SSD cost defined in (6) with the Normalized Cross-Correlation (NCC) matching cost defined as:

$$M_{ncc}(l, r) = \frac{1 - M'_{ncc}(l, r)}{2}$$

where

$$M'_{ncc}(l, r) = \frac{\sum_{\epsilon \in \Omega} (I_{\mathbf{p}_{l+\delta}}^l - \bar{I}_{\mathbf{p}_l}^l)(I_{\mathbf{p}_{r+\delta}}^r - \bar{I}_{\mathbf{p}_r}^r)}{\sqrt{\sum_{\epsilon \in \Omega} (I_{\mathbf{p}_{l+\delta}}^l - \bar{I}_{\mathbf{p}_l}^l)^2 \sum_{\epsilon \in \Omega} (I_{\mathbf{p}_{r+\delta}}^r - \bar{I}_{\mathbf{p}_r}^r)^2}} \quad (6)$$

is the *correlation coefficient*. We have found little difference, in terms of results, between the two implementations but SSD is considerably faster than NCC (despite our efforts to improve the efficiency of the NCC code⁴).

⁴www.idiom.com/~zilla/Work/nvisionInterface/

5.1 Inter-scanline consistency and cost aggregation

A solution to the issue of inter-scanline consistency [12] is to propagate information across scanlines by detecting and matching vertical edges. This has two drawbacks however: (i) the robust matching of edges is an open issue, especially for occluding contours (precisely where we need most accuracy); (ii) edge detection and matching algorithms are slow. Our solution to the problem of encouraging the propagation of information across scanlines efficiently is to use small window neighborhoods in the cost computation step (3×3) followed by a separate cost aggregation step. The aggregation acts directly on the matching cost function, and so is different from simply post-processing the output path, or smoothing the output disparity and occlusion maps.

The algorithm proceeds as follows: Firstly the cost matrices $M(l, r)$, associated with each pair of scanlines, are built and stacked to form a three-dimensional cost space as in fig. 14a. Secondly, an anisotropic 2D Gaussian smoothing filter is applied with principal axes \mathbf{a} parallel to the virtual image plane (fig. 14b). The axis \mathbf{a} of the Gaussian kernel orthogonal to the left and right scanline axes is responsible for enforcing inter-scanline consistency of the costs; the other axis \mathbf{a}' produces additional smoothing of sharp corners in the occlusion map by encouraging fronto-parallel surfaces [15]. Typical values for Gaussian smoothing parameters are: $\sigma_a = 3$ pixels along the \mathbf{a} axis and $\sigma_{a'} = 2$ pixels along the \mathbf{a}' axis. The key property of the cost-filtering process is that it manages to propagate cost information across scanlines without smoothing the final path or the disparity map. In fact, cost filtering precedes DP optimal path finding. Furthermore, the cost filtering step, being a simple 2D convolution (separable), can be implemented efficiently by using two 1D convolutions.

Notice that if we had used standard, un-normalized SSD in the cost computation step, then the use of large window neighborhoods (with Gaussian weighting) would have been equivalent to the Gaussian cost aggregation performed in this section. However, the use of normalized matching scores makes the matter more complicated. Furthermore, we have found that Normalized SSD costs on small windows work considerably better than standard SSD. Further research is necessary here to assess an optimal matching cost function.

The output of the cost aggregation process is the new set of $M(l, r)$ costs used, as input to the improved DP algorithm described in section 4.

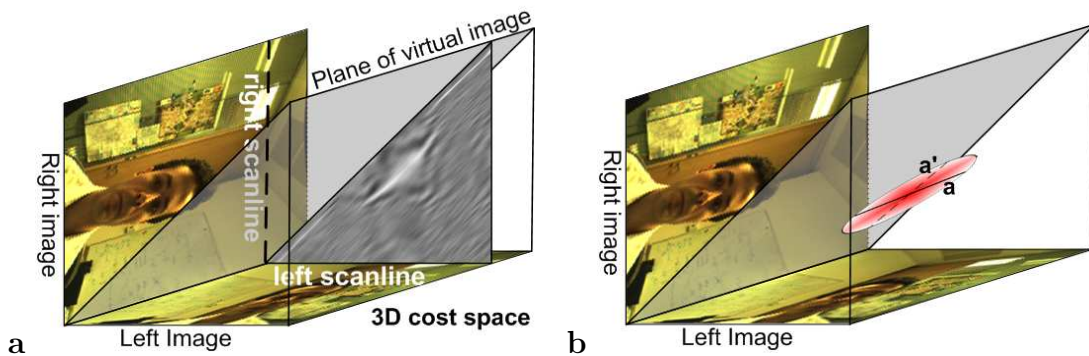


Figure 14: The 3D cost space for a pair of stereo images. (a) The basic diagram of fig. 5a becomes a 3D diagram when all the scanline pairs are considered. The value of each element inside the parallelepiped is the $M(l,r)$ cost of matching two pixels. The objective of DP is that of finding the minimum-cost surface inside this 3D cost space. The diagonal plane is called the “plane of virtual image” for reasons that will become apparent in the remainder of the paper. (b) In order to propagate cost information across scanlines a 2D Gaussian filter (represented by the red ellipse) parallel to the virtual image plane is applied to the 3D cost space.

6 Evaluating recovered occlusion maps

The goal of this section is two-fold: i) demonstrating the advantages of our new DP algorithm with respect to conventional DP and, ii) defining measures of accuracy of the occlusion maps for comparison with state of the art Graph-Cut techniques.

6.1 The advantages of the 4-move, 4-plane model

Four moves vs three moves Figure 15 demonstrates the effect of moving from the standard three-move DP algorithm to the new four-move one.

Comparing fig. 15a and fig. 15b one can see that the four-move model removes most of the incorrect occlusion events which occur in the background of fig. 15a (isolated red and green points). The black pixels in fig. 15a,b correspond to matched moves. The four-way model correctly classifies small jumps in disparity levels as matched moves discretizing slanted surfaces (*e.g.* the face or the slanted walls in the background).

Four planes vs one plane The real improvement is obtained when the four-move model is married to the four-plane graph. As stated previously, this new graph structure is used to favour long runs of occlusions. In fig. 15c the occlusions are: (i) correctly located along

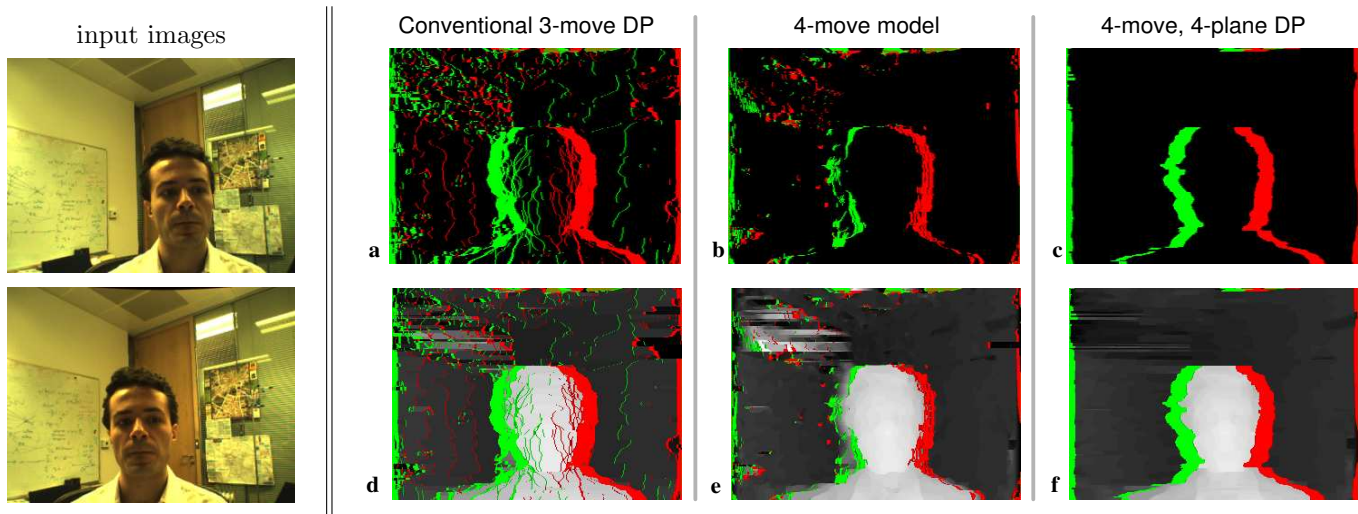


Figure 15: Comparing the different DP models. *(a,b,c) Occlusion maps obtained from three different DP algorithms. Red indicates left occlusions and right occlusions are green. (a) Standard 3-way DP applied to the stereo pair on the top row. Numerous matched pixels are ambiguously classified as possibly occluded. True occlusions around the head show as broken up maps of ambiguous labels, and would cause “reset” artefacts in the cyclopean view. (b) The simple four-move model (section 3). Many of the spurious occlusion labels have disappeared. The occluded areas around the head still show some fragmentation. (c) The real improvement is achieved when the four-move model is married with the four-plane graph: note the unfragmented occlusion regions adjacent to the head. Throughout, 7×3 window patches were used in the computation of matching costs; a value of $\sigma = 2.0$ has been used for the cost filtering step. (d,e,f) The recovered disparity maps, corresponding to (a,b,c), respectively: removal of spurious occlusion labels helps also to clean up the disparity map. The disparity values have been scaled up 1.5 times for visualization purposes.*

the boundary of the foreground object, and (ii) detected as compact, solid regions. This, in turn, leads to better occlusion maps and more convincing synthesis, as will be demonstrated in section 9.

Inter-scanline consistency Figure 16 demonstrates the effect of inter-scanline information propagation by cost filtering. It can be observed that the cost filtering step increases the quality of the labelling of occlusion. In particular, in our algorithm, as the value of the standard deviation σ_a of the Gaussian kernel increases the runs of occlusions become correctly aligned. Furthermore, above a certain value of σ_a the results become quite stable.

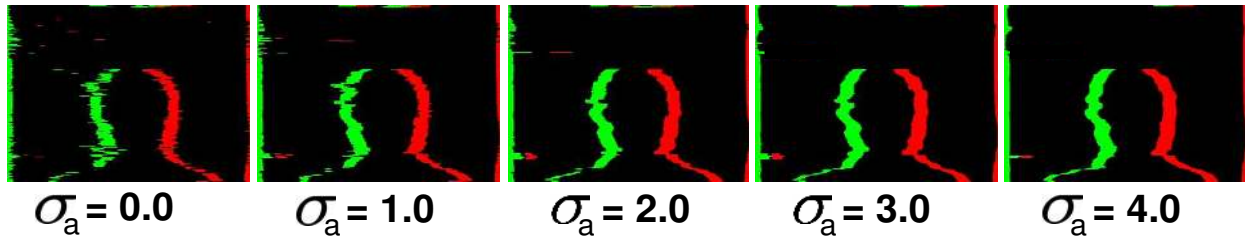


Figure 16: Inter-scanline consistency by cost filtering. Occlusion maps obtained by 4-move, 4-plane DP for different values of σ_a in the cost Gaussian filtering step (section 5.1). In this experiment the value of $\sigma_{a'}$ has been kept fixed at $\sigma_{a'} = 2$ pixel. This figure illustrates how cost filtering helps achieve more “compact” occlusion labels.

6.2 Estimating the occlusion map of a standard test stereo pair.

Figure 17 illustrates the results of applying our algorithm to a pair of images from one the standard test sets of Szeliski and Scharstein⁵. In this example the input images are characterized by a maximum disparity of about 29 pixels, and a maximum occlusion gap of 24 pixels *i.e.* 8.5% of the image width (image dimensions are 284×216).

The results of estimating occlusion and disparity maps (fig. 17c,d) are quite convincing. However, our algorithm has been designed to cope with situations involving much greater disparity ranges than the ones shown in [15]. For the next experiment we have created our own test stereo pair, characterized by a much larger disparity and occlusion range.

6.3 Comparisons with state of the art.

Figure 18 compares our results with the ones obtained by three well known graph-cut techniques [2,9,10] and the conventional 3-move DP [5]. Excepting the graph-cut method in [10]⁶, the other algorithms are based on our own implementations.

Figure 18a,b are the two input images used in this experiment. The photographed scene is made of two background slanted planes and one foreground front-on plane. The stereo pair is characterized by a maximum disparity range of 90 pixels and a maximum occlusion gap of 72 pixels which corresponds to 22.5% of the image width (image dimensions are 320×240); almost *three* times the disparity of the standard test data set in fig. 17. The ground truth

⁵<http://www.middlebury.edu/stereo/>

⁶Original algorithm available from

www.cs.cornell.edu/People/vnk/software.html

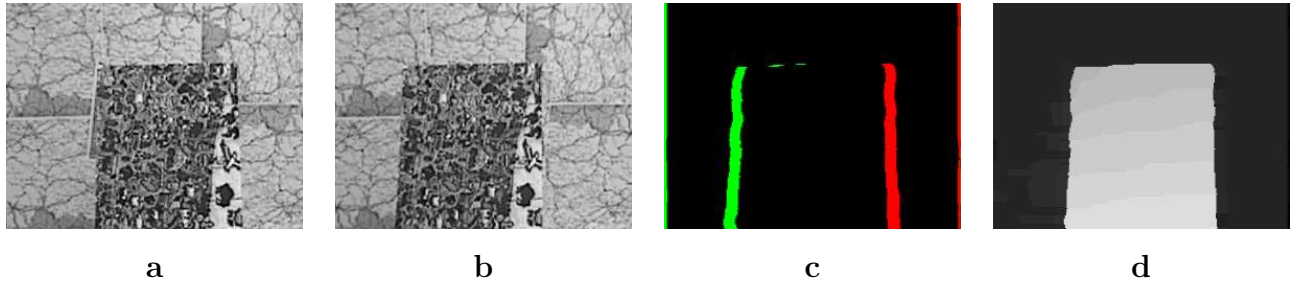


Figure 17: Estimation of occlusion and disparity maps from the Map standard test stereo pair. (a,b) *Input left and right images, respectively. The maximum occlusion gap is about 24 pixels, i.e. 8.5% of the image width.* (c) *Recovered occlusion map for the cyclopean view;* (d) *Estimated disparity map for the cyclopean view. The disparity values in the occluded areas have been synthesized by the static occlusion filling algorithm discussed in section 2.*

disparity and occlusion map (fig. 18c) was obtained by least-square fitting of the two planes in the background and the planar surface of the foreground object. The fitting process was initialised by an accurate and dense, pixel-wise match produced by our DP algorithm. The segmentation of the foreground object was performed manually and the correctness of the resulting ground truth was verified by manual inspection. Some graph-cut algorithms such as [10] produce left and right occlusion maps and not the cyclopean map. Therefore, in order to reduce the possibility of error we have decided to compare the performance of the selected algorithms always with reference to the left camera. Fig. 18d-h show the results of computing the left-referenced occlusion and disparity maps via different algorithms.

In order to quantify the accuracy of occlusion detection we define two measures: i) the *Misclassification rate* and ii) the *Isolation rate*. The former is a function of the number of wrongly classified pixels, while the latter measures the degree of *compactness* of the occluded and matched regions in the occlusion maps.

Misclassification rate is estimated by comparing the occlusion maps recovered by each algorithm with the ground truth (fig. 18c) and counting the number N_m of misclassified pixels (both false positives and false negatives).

Isolation rate The isolation rate is the proportion of pixels that are isolated — those for which the number of neighbouring pixels with the same label (matched or unmatched) is lower than a predefined, fixed threshold ξ .

Formally: given a point \mathbf{p} in the occlusion map, its classification label $L(\mathbf{p})$ with $L(\mathbf{p}) \in$

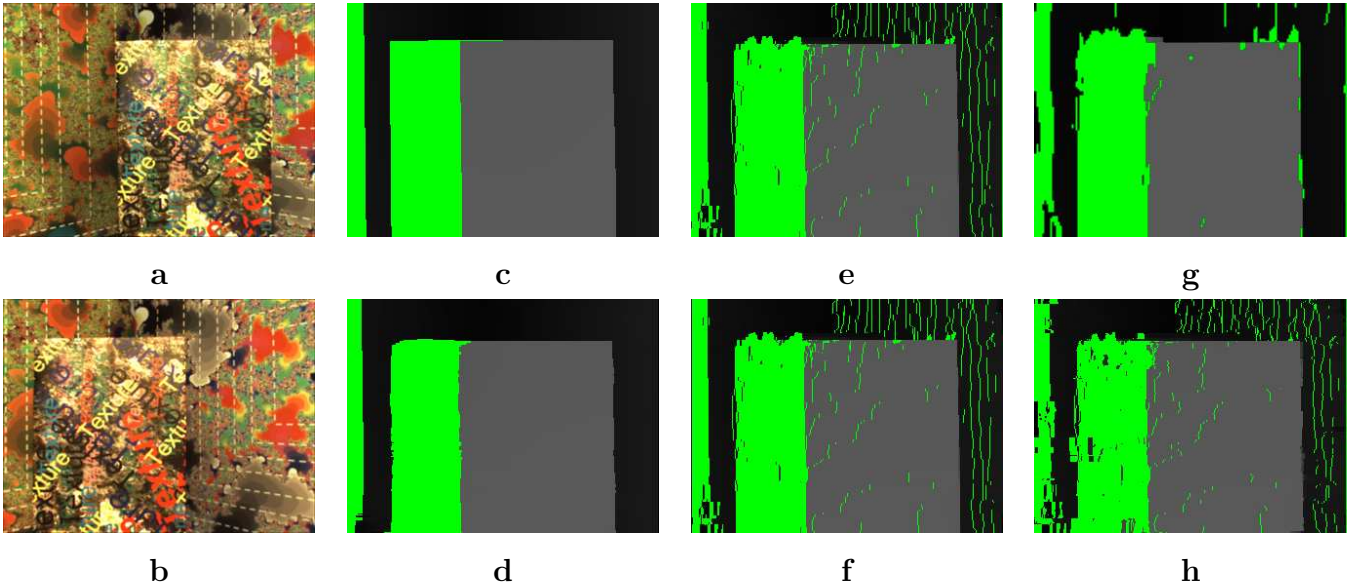


Figure 18: Comparing the occlusion maps returned by different algorithms. (a,b) Input left and right images, respectively. (c) Ground-truth occlusion map with reference to the left camera. As usual, green indicates right-occlusion. The maximum occlusion gap is about 72 pixels, i.e. 22.5% of the image width; much larger than the occlusion in fig. 17. (d-h) Left-referenced occlusion maps recovered by using: (d) our 4-plane, 4-move DP algorithm; (e) Buehler et al. graph-cut algorithm [2]; (f) Kolmogorov et al. graph-cut algorithm [10]; (g) Ishikawa et al. graph-cut algorithm [9]; (h) Cox et al. dynamic-programming algorithm [5].

$\{Matched, Occluded\}$ and a neighborhood window Ψ_p centred in \mathbf{p} with size $W \times W$, then we define N_p as the number of points $\mathbf{q} \in \Psi_p | L(q) = L(p)$. The pixel \mathbf{p} is classified as “isolated” if $N_p < \xi$. In our experiments we used $\xi = 2$ and $W = 3$. Unlike the misclassification rate, the isolation rate does *not* rely on ground truth data.

Comparative results The misclassification rate and the isolation rate have been measured for all the occlusion maps in fig. 18d-h and the results shown in Table 2. Notice that the three graph-cut algorithms [2, 9, 10] perform comparably well and considerably better than standard three-move DP. These results, although derived from comparing occlusions rather than disparities, are consistent with those obtained by Scharstein and Szeliski [15]. The reduced misclassification and isolation rates obtained by our new DP algorithm are due to the extended four-label pixel classification and the enforcement of occlusion-run constraints. While the GC framework in [10] supports runs of occlusions, these are not correctly

<i>Algorithm</i>	<i>Misclass. rate</i>	<i>Isolation rate</i>	<i>Runtime</i>
Our algorithm	2.61%	0.005%	1.57s
Buehler <i>et al.</i> [2]	6.45%	0.361%	468 s
Kolmogorov <i>et al.</i> [10]	6.57%	0.367%	65 s
Ishikawa <i>et al.</i> [9]	6.61%	0.218%	912 s
Cox <i>et al.</i> [5]	8.17%	0.467%	0.31 s

Table 2: Comparisons with state of the art. Comparing accuracy and performance of different dense stereo algorithm in estimating occlusion maps.

modeled in the sense that occluded moves are used to approximate slanted surfaces. Furthermore, right and left occlusions are not differentiated. An alternative GC algorithm [9] that does use explicit labels for occlusion produces poor results for lack of constraint enforcement. The effect of approximating slanted surfaces with occluded moves can be observed in figs. 18e,f,h; where the background is constellated by a large number of vertically aligned occluded pixels (marked in green). Thus, these results show that the combination of both an extended occlusion model for correct pixel classification and the enforcement of constraints on occluded areas achieves the best results. Table 2 also shows our algorithm being the second fastest, immediately after the very efficient (but poor quality) Cox DP.

Further notes on our experimental procedure It must be stressed that the different energy minimization algorithms analysed in this section have been applied to *exactly* the same cost space, which was computed only once⁷. This was done to eliminate variability due to different matching cost functions or cost smoothing parameters. Furthermore, for each algorithm we have selected the combination of parameters which has lead to the best results for that specific algorithm. In the case of the algorithm in [10] the parameters were automatically selected by the original implementation. Finally, all algorithms were run on the same machine, a 3GHz, 1Gb RAM Pentium IV desktop computer. In contrast to [15], our results re-instate dynamic-programming techniques amongst the most accurate and efficient ones for shape recovery from large-disparity image stereo pairs. Furthermore, Table 2 suggests two more occlusion-based error metrics which should be added to the set

⁷Note that we had to adapt the source code in [10] to read our filtered cost space as input. Then, graph-cut was used for energy minimization only.

of metrics defined in [15].

The previous sections have: i) stressed the importance of accurate occlusion detection, ii) illustrated our algorithm for the reliable estimation of occlusion and disparity maps and iii) evaluated our algorithm against state of the art techniques. The next sections, instead, focus on the new-view rendering problem, *i.e.* how to best make use of the extracted geometric information for the purposes of virtual-image synthesis.

7 Rendering occlusions

As described in section 2.2, in order to produce high-quality virtual images, it is necessary to synthesize the occluded regions in a robust and realistic way. We have investigated two strategies for occlusion filling: *static* filling, which applies to single pairs of stereo images and *temporal* filling which, instead, models what lies behind the occlusions from long sequences of stereo images.

Static occlusion filling Given an input stereo pair of images 'fronto-parallel' synthesis of the occluded regions is done via the algorithm illustrated in fig. 6c. As discussed in section 2.2 an effective filling of the occlusions is achieved only for solid and correct occluded regions. Figure 19b demonstrates that by applying our improved DP algorithm for the estimation of the minimum-cost surface and the occlusion maps, followed by the static occlusion filling with fronto-parallel assumption, correct-looking synthesis in the occluded areas is achieved. Notice that fig. 19b is free from any "halo" or "reset" artefacts.

Temporal occlusion filling When the static filling algorithm is applied to long image sequences, temporal artefacts become visible in occluded regions. Moreover, because of the lack of pixel correspondence, stability of synthesis is a particular issue in the occluded areas. One solution to this problem is the construction and dynamic update of a model of the background used to fill in the regions of missing information. The algorithm is in two steps: the first step segments the foreground from the background at each time instance; the second step uses the newly discovered (disoccluded) pixels of the background to improve the background model.

The segmentation step, performed at each time instance, proceeds as follows:

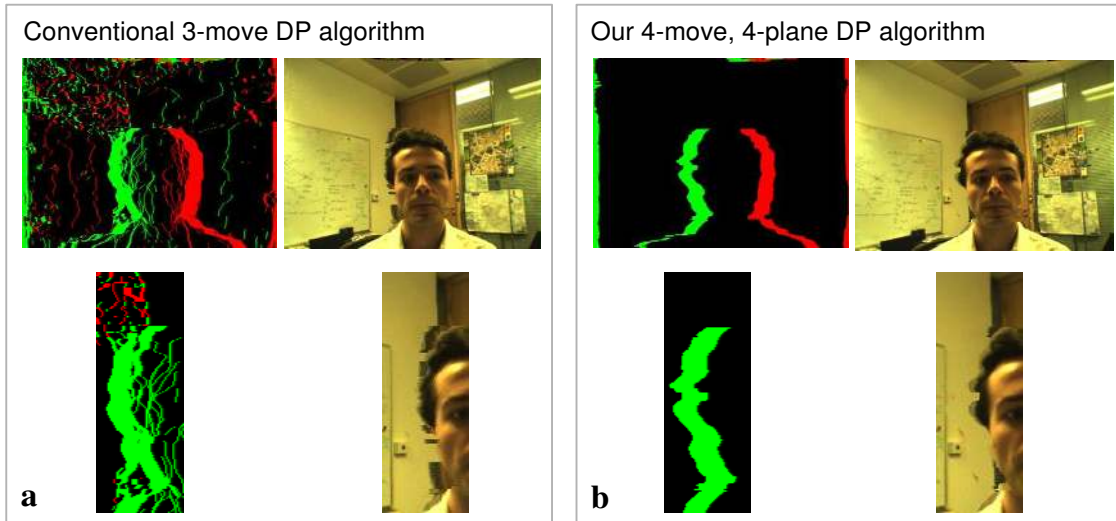


Figure 19: Eliminating the reset artefact. (a) Occlusion map and reconstructed cyclopean view for conventional DP. Small islands of spurious matched pixels inside occlusion regions cause the reconstruction of the occluded areas to fail. Notice also that many pixels on slanted surfaces have been incorrectly classified as occluded. (b) Occlusion map and reconstructed cyclopean view for the proposed four-move, four-plane DP algorithm. The compactness of the recovered occlusion regions produces a much nicer cyclopean reconstruction: the background door frame is now straight and almost completely artefact-free.

Given the estimated min-cost surface \mathcal{S} :

1. Along each scanline in the min-cost surface, for each run of occlusions, the disparity at the highest disparity end of the run is histogrammed (fig. 20).
2. The valley in the resulting bi-modal histogram determines the adaptive threshold disparity value \hat{d} that is used for the background/foreground segmentation.

Figure 20b shows a typical histogram. The peak near the origin is due to the long and thin occlusion bands at the edges of the image, while the peak at higher disparity values is due to the foreground object and is the one we are mostly interested in. This kind of bi-modal histogram turns out to be characteristic of a large number of sequences of talking heads. This approach for automatic threshold detection works better than histogramming the whole set of estimated disparities. This is because the selected pixels (marked in white in fig. 20a) are more representative of the foreground object. This technique has been proven to work also in situations where part of the background are very close (in depth) to the

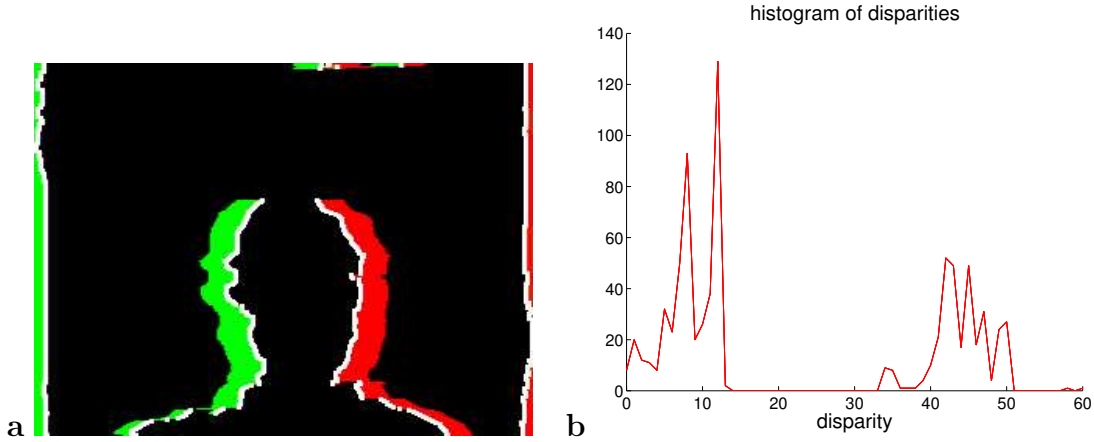


Figure 20: Foreground/background segmentation. (a) The pixels corresponding to the higher-disparity end in each horizontal run of occlusion is marked in white. (b) The histogram corresponding to the disparities extracted in (a).

talking head (*e.g.* a receding wall). In the second step of the algorithm a background model is constructed and updated at each time instance. The background model is made of three elements: its disparity map D_B in cyclopean coordinates, and the corresponding left and right images I_B^l and I_B^r , respectively. At each time instance t the background model is updated by the following rule:

$$\begin{aligned}
 D_B^t(\mathbf{p}) &= \phi D_B^{t-1}(\mathbf{p}) + (1 - \phi) D^t(\mathbf{p}) \\
 I_B^l(\mathbf{p}_l) &= \phi I_B^{l,t-1}(\mathbf{p}_l) + (1 - \phi) I^l(\mathbf{p}_l) \\
 I_B^r(\mathbf{p}_r) &= \phi I_B^{r,t-1}(\mathbf{p}_r) + (1 - \phi) I^r(\mathbf{p}_r)
 \end{aligned} \tag{7}$$

where \mathbf{p} is a pixel whose disparity $D(\mathbf{p})$ falls below the automatically computed foreground threshold \hat{d} (and thus belongs to the background). The points \mathbf{p}_l and \mathbf{p}_r are the corresponding positions on left and right input images, respectively. $D_B^t(\mathbf{p})$ is the disparity of the pixel \mathbf{p} in the current background model at time t . The scalar factor ϕ represents a decay constant ($0 \leq \phi \leq 1$). The update rule in (8) applies to all the pixels which belong to the background and are visible and does not apply to occluded pixels. The use of the exponential memory parameter ϕ allows for a relaxation of the static background assumption. In our experiments $\phi = 0.9$ achieves a good balance between keeping the previous values of the background pixels and updating them in the case of dynamic events on the background.

Figure 21 illustrates the results of the temporal background filling algorithm. During the video-communication session the head moves and disoccludes portions of the background.







frame number	0	70	170
synth. cyc. image			
background model			

Figure 21: Temporal background generation. (top row) Synthesized cyclopean views for different frames. More examples of synthesised cyclopean views are provided in the results section. (bottom row) Corresponding background models. As new regions of the background are discovered the background model is updated and the blank region (occlusion) progressively filled.

The background model is updated and, after a few frames, if the head moves substantially, the background is completely reconstructed.

Advantages and disadvantages of static and temporal filling strategies The static occlusion filling strategy is based on the assumption of a fronto-parallel background, which, although most of the time produces good results, may not make sense for scenes with very slanted surfaces. Furthermore, the static occlusion filling requires solid and accurate occlusion areas which are achieved by our four-plane DP algorithm but are not in conventional DP or graph-cut techniques. On the other hand, in the temporal occlusion filling, the background model is learnt from the disocclusions of previous frames. This introduces the need for background/foreground segmentation and the assumption of a quasi-static background.

Temporal occlusion filling and background modeling is especially useful in the next section which introduces the three-dimensional motion of the virtual camera. In fact, as the virtual camera centre moves away from the baseline of the two input cameras less information is available from the current pair of stereo frames about the occluded regions, and temporally acquired background information becomes extremely useful for reconstructing unseen regions. Overall, we have found that a combination of the two techniques works best: we

use static filling in the half-occluded areas which have not yet been observed, and temporal filling in those occluded regions which have been disoccluded in previous frames.

8 Rendering from variable viewpoint

The ability to create virtual images from generic viewpoints is of considerable interest both for interactive video and teleconferencing applications. Conventionally, one way of generating novel views from virtual camera locations is by: (i) transforming the computed disparity map into a 3D surface (*e.g.* by means of a triangle mesh), (ii) texture mapping it with one of the two images, (iii) projecting the texture-mapped surface into the plane of the virtual camera. This section describes a novel, compact technique for rendering virtual views *directly* from the estimated disparity surface, thus overriding the need to construct an explicit 3D model of the scene.

The geometry of the virtual camera Figure 22 shows a plan view of the system with the optical centre of the virtual camera being placed in the generic location denoted \mathbf{O}_v . A 3D scene point \mathbf{P} is projected on the left and right images into the points $\mathbf{p}_l = (x_l, y_l)^\top$ and $\mathbf{p}_r = (x_r, y_r)^\top$, respectively. Also, \mathbf{P} is projected on the cyclopean camera (with optical centre in $\mathbf{O}_c = \mathbf{O}$) in the point $\mathbf{p}_c = (x_c, y_c)^\top$ (not shown in the figure) and on the virtual camera in the point $\mathbf{p}_v = (x_v, y_v)^\top$. The disparity between the corresponding left and right image points is easily computed as

$$d = x_l - x_r = f \frac{B}{Z} . \quad (8)$$

In the cyclopean camera, by triangle similarity we can compute

$$x_c = f \frac{X}{Z} . \quad (9)$$

For a virtual camera with optical center in $\mathbf{O}_v = (T_x, T_y, T_z)^\top$ we can write: $(X - T_x) : x_v = (Z - T_z) : f$, from which

$$x_v = f \frac{X - T_x}{Z - T_z} . \quad (10)$$

By substituting (8) and (9) into (10) we obtain: $x_v = \frac{x_c - dT_x/B}{1 - dT_z/(fB)}$ which, together with the

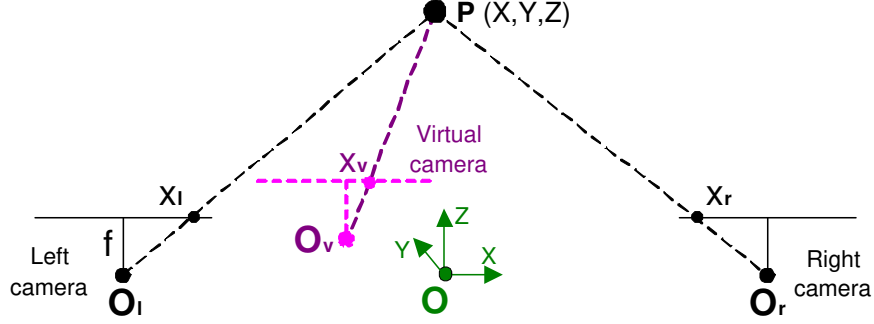


Figure 22: Basic notation for virtual image generation. \mathbf{O}_l , \mathbf{O}_r and \mathbf{O}_v are the optical centres of left, right and virtual cameras respectively. The optical centre of the virtual camera can be placed anywhere in space and the corresponding virtual image is synthesized by our algorithm.

analogous equation for the y_v coordinate, can be rewritten in homogeneous coordinates as:

$$\begin{pmatrix} x_v \\ y_v \\ w \end{pmatrix} = \begin{bmatrix} 1 & 0 & -T_x/B & 0 \\ 0 & 1 & -T_y/B & 0 \\ 0 & 0 & -T_z/(fB) & 1 \end{bmatrix} \begin{pmatrix} x_c \\ y_c \\ d \\ 1 \end{pmatrix} \quad (11)$$

Equation (11) represents a projection of 3D points into a plane [8]. It can be proven that (11) corresponds to projecting points of the min-cost surface into the corresponding points on the plane of the virtual image (up to a scale, diagonal matrix) as illustrated in fig. 23a.

From (11) the centre of projection \mathbf{Q} is readily computed as the null vector of the projection matrix, thus yielding: $\mathbf{Q} = \left(\frac{T_x}{B} \quad \frac{T_y}{B} \quad 1 \quad \frac{T_z}{fB} \right)^\top$. Notice that for $T_z = 0$ the transformation (11) is a *parallel* projection (\mathbf{Q} is at infinity). This, in turn means that sidewise motion (in the X direction) and up/down motion (in the Y direction) of the virtual camera can be easily simulated by projecting points of the disparity surface \mathcal{S} onto the virtual image plane via parallel rays. On the contrary, the inwards/outwards translation of the virtual camera ($T_z \neq 0$) is achieved by means of a *central* projection with *finite* centre of projection \mathbf{Q} . The simple mapping between the motion of the centre of projection \mathbf{Q} and the corresponding translation of the virtual camera is illustrated in 23c,d. For instance, inwards camera translation (not zoom) is achieved by moving the centre \mathbf{Q} from $+\infty$ towards the plane of the virtual image.

Notice that for $\mathbf{Q} = (-1/2, 0, 1, 0)^\top$ (i.e. $\mathbf{O}_v = (-B/2, 0, 0)^\top$) the virtual image corresponds to the input left image, for $\mathbf{Q} = (1/2, 0, 1, 0)^\top$ (i.e. $\mathbf{O}_v = (B/2, 0, 0)^\top$) the virtual image corresponds to the input right image, and for $\mathbf{Q} = (0, 0, 1, 0)^\top$ (i.e. $\mathbf{O}_v = (0, 0, 0)^\top$)

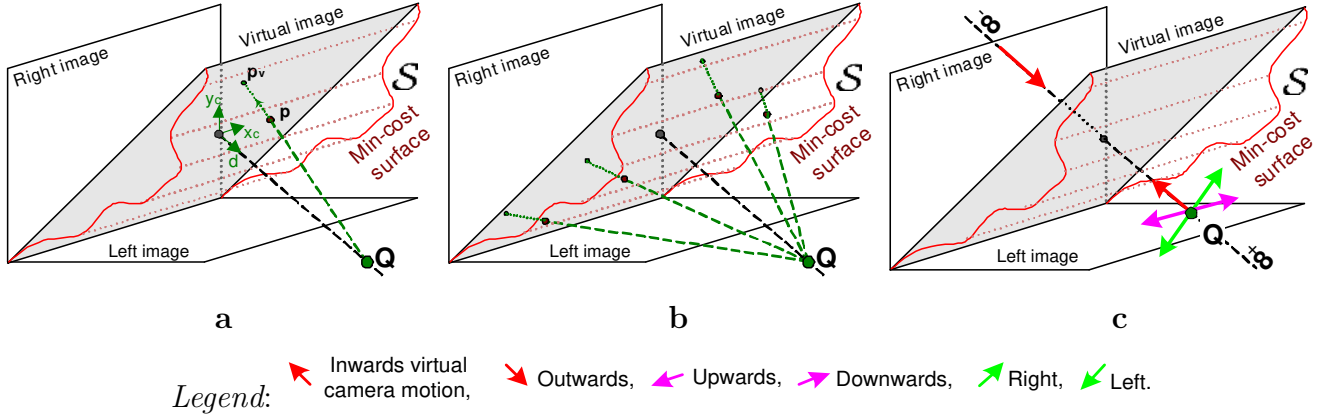


Figure 23: Virtual camera motion. (a) The 3D motion of the virtual camera is achieved by direct projections of points on the minimum cost surface into the virtual image plane. The reference coordinate system (x_c, y_c, d) has origin in the centre of the virtual image plane. (b) The virtual image is generated directly by projecting points from the minimum-cost surface S into the virtual image plane. (c) Moving the centre of projection Q corresponds to translating the virtual camera. The coloured arrows indicate the mapping between moving the centre of projection Q in our diagram and the corresponding translations of the virtual camera in the scene.

the virtual image corresponds to the cyclopean image.

Synthesizing virtual images from generic viewpoints Given a point \mathbf{p} on the minimum-cost surface and its corresponding virtual position \mathbf{p}_v (fig. 23a and cf. fig. 6a), the corresponding pixel value (intensity or colour) is given by a combination of the pixel values of the corresponding pixels \mathbf{p}_l and \mathbf{p}_r in the input images according to the following equation⁸:

$$I^v(\mathbf{p}_v) = (1 - \mu)I^l(\mathbf{p}_l) + \mu I^r(\mathbf{p}_r) \quad (12)$$

with $\mu = \frac{|\mathbf{O}_x^v - \mathbf{O}_x^l|}{B}$; where the subscript indicates the x component of optical centres of the two input cameras.

Occlusion filling and rendering The filling of occlusions for generic virtual view placement is very similar to the cyclopean case illustrated in fig. 6c. As shown in fig. 24 now the direction of projection is dictated by the position of the centre Q , the cyclopean case being a special case of this general projection.

⁸Equation 12 is strictly valid only for matched pixels; while values of occluded pixels are taken only from the image where they are visible (fig. 6b).

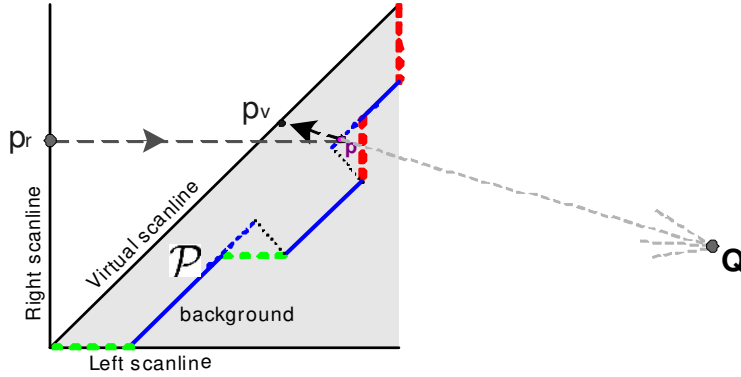


Figure 24: Occlusion filling for generic virtual camera placement. *The only difference with respect to the cyclopean occlusion filling illustrated in fig. 6c is that now the generic direction of projection is dictated by the position of the centre Q .*

The rendering algorithm described here is an extension of the cyclopean rendering presented in section 2.2. By inspection of (12) one can see that in the cases where $O_v = O_l$ or $O_v = O_r$, the original left and right views are resynthesised *exactly*, and independently from the recovered disparities, as expected. Further advantages of our rendering technique are: (i) direct view-dependent texture rendering which negates the need for surface triangulation and (ii) effortless occlusion reconstruction by simple projection of the minimum-cost surface. High-quality output images are obtained by standard reverse mapping and bilinear interpolation techniques. Notice that rotations of the virtual camera have not been considered here. Rotations may be achieved by homography-based image warping. However, virtual-camera rotation does not seem to be an important requirement in video-conferencing.

9 Results: novel-view synthesis

This section presents a number of synthesis results achieved on real input sequences. In particular, we demonstrate: gaze correction, cyclopean view generation, three-dimensional translation of the virtual camera, simple augmented-reality effects such as object insertion and replacement.

Gaze correction by cyclopean view synthesis for still stereo images Figure 25 shows an example where the input left and right images of fig. 3 have been used to generate the cyclopean view via the proposed algorithm. Notice that the spatial artefacts (streaks

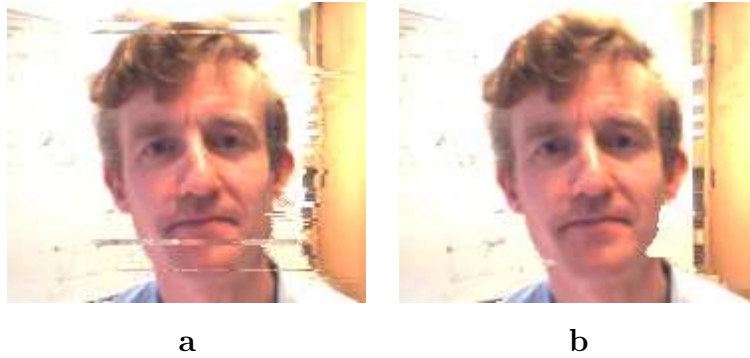


Figure 25: Example of gaze correction. (a) Cyclopean image synthesized via the algorithm in [4]. This image is identical to that in fig. 3b and is repeated here for clarity. The input left and right images are shown in fig. 3a,c, respectively. (b) The cyclopean, gaze-corrected view generated by our algorithm. The gaze has been corrected while eliminating the artefacts of (a).

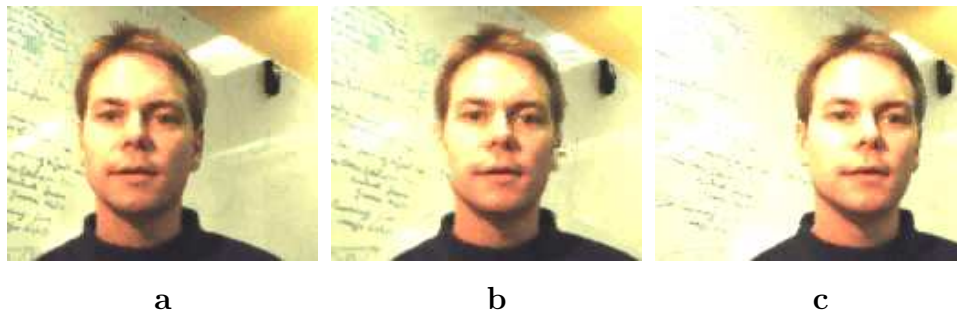


Figure 26: Another example of gaze correction. The central image, (b) has been generated from the two input views (a,c) and shows correct gaze (the person is looking at us). There are no significant “halo” effects or streaky artefacts.

in fig. 3b) have been removed. In the output image (fig. 25) the gaze has been corrected. Another example of gaze correction from static images is illustrated in fig. 26.

3D translation of the virtual camera Figure 27 shows an example of translating the virtual camera towards and away from the visualized scene. Notice that this is different from simple zooming or cropping of the output image. In fact, parallax effect may be noticed in the boundary between the head and the background, thus providing the correct three-dimensional feel.

Figure 28 shows an example of in-plane translation (with \mathbf{O}_v on the XY plane) of the virtual camera. Notice the relative displacement of the head with respect to the background.

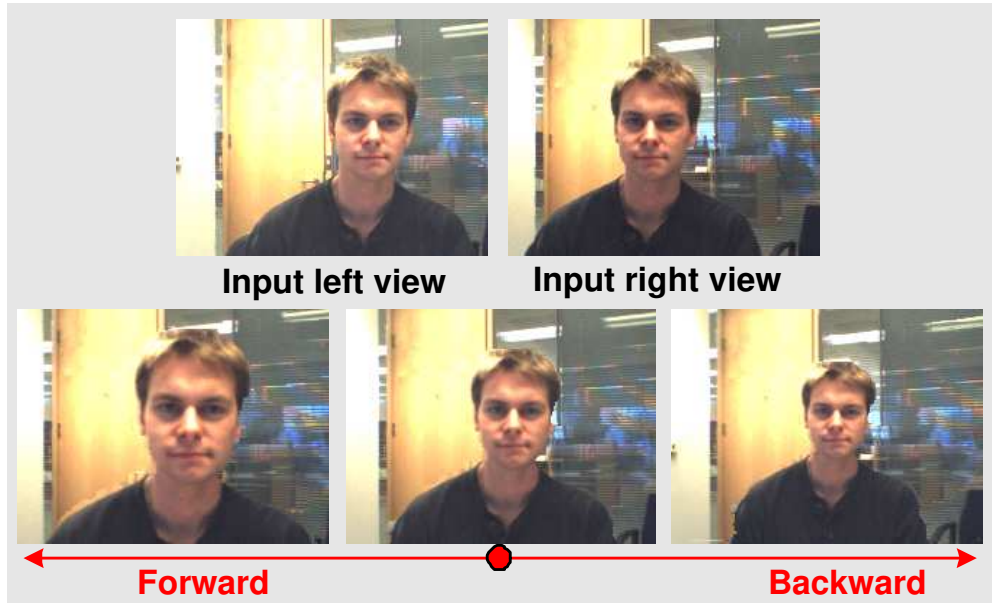


Figure 27: Forward/backward translation of virtual camera. *The bottom row shows the synthesized cyclopean views with (left) forward virtual camera translation, (centre) no virtual camera translation, (right) backward virtual camera translation. Notice the parallax effect around the head.*

Cyclopean view generation in long sequences Figure 29 demonstrates the effectiveness of the proposed algorithm for reconstructing cyclopean views of extended temporal sequences. It can be observed that most of the spatial artefacts (*e.g.* streaks, halo) and temporal artefacts (*e.g.* flickering) are removed.

Basic 3D scene editing The proposed algorithm generates novel, virtual views, but also, as a by-product, a 3D representation of the observed scene. The latter can be advantageous for 3D scene editing. As a first example, fig. 30 demonstrates the possibility of replacing the original background with a different one, either taken from real photographs or artificially generated. This is made possible thanks to the foreground/background segmentation step described in section 7. Sophisticated matting techniques for high-quality layer compositing are not the focus of this paper.

Finally, fig. 31 demonstrates the possibility of inserting three-dimensional emoticons during the teleconferencing session. The geometric understanding of the viewed scene allows the emoticons to be realistically animated in the three-dimensional space. Partial and total occlusion events are correctly handled.

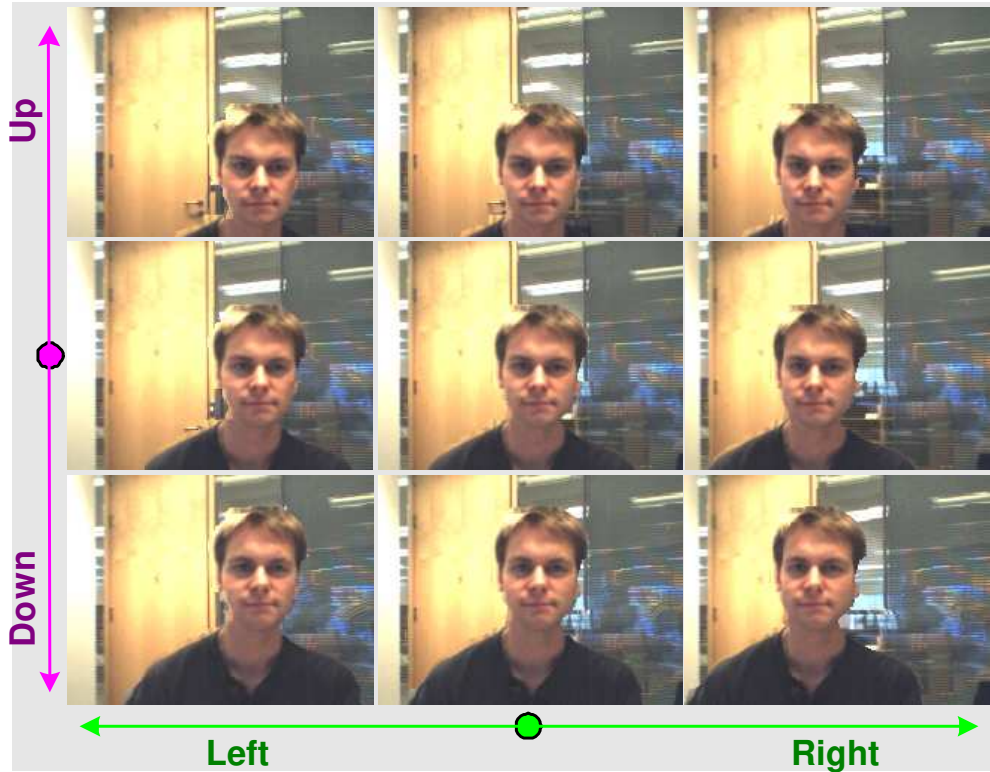


Figure 28: In-plane translation of virtual camera. *The left and right input images are the same as in fig. 27. This table shows the synthesized images corresponding to translation of the virtual camera along the x and y axes. Notice the parallax effect around the head. Also, the door frame is reconstructed nicely despite it being partially occluded in the right input view.*



Figure 29: Cyclopean image synthesis for long sequences. *Frames extracted from a reconstructed cyclopean sequence (over 10 sec long). The input images are not shown here.*

10 Conclusions and future work

This paper has described an efficient algorithm for the synthesis and geometric manipulation of high-quality virtual images generated from a pair of synchronized stereo sequences with large disparities. In this paper we have focused on one-to-one teleconferencing applications but the techniques are more general and can be employed in other fields requiring high-quality novel view generation and dense stereo.

The main contributions of the paper can be summarized as:

- A new four-label, four-plane DP algorithm for the correct detection and classification of occlusion events;
- Anisotropic filtering in the cost space for information propagation across scanlines;
- An elegant and compact geometric technique for the rendering of novel views *directly* from the minimum-cost surface estimated by the DP algorithm.



Figure 30: Background replacement. *The techniques developed in this paper allow, amongst other things, for the foreground to be segmented from the background. This, in turn, allows the real background to be replaced by more interesting and prettier images, or video-textures as shown in this case.*

The effectiveness of the new algorithmic components has been demonstrated in a number of examples where the artefacts typical of DP techniques have been eliminated while keeping quite a high frame rate. The current implementation exploits SSE2 instructions and produces virtual images at about 7 frames per second (on 320×240 images, on a 3.0Ghz Pentium IV with 1Gb RAM). We are planning to improve on the algorithm efficiency in order to obtain as reliable results in real time. The viability of the proposed algorithm has also been demonstrated by comparing the accuracy of the estimated occlusion maps with the ones generated by state of the art techniques amongst which three of the most recent graph-cut algorithms.

Currently we are investigating the use of other possible cost functions; we are running more rigorous comparisons with graph-cut techniques and experimenting with different cost filtering techniques. Finally, thorough experimentation with different camera layouts and the generation of standard test sequences for evaluation will be undertaken.

Acknowledgements The authors would like to thank G. Smyth, G. Cross, V. Kolmogorov, I. Cox, D. Scharstein, R. Szeliski, Y. Boykov, for their useful comments and inspiring discussions.

References

- [1] P.N. Belhumeur and D. Mumford. A Bayesian treatment of the stereo correspondence problem using half-occluded regions. In *IEEE Comp. Soc. Conf. on Comp. Vision and*



Figure 31: Inserting 3D emoticons. *The proposed algorithm generates depth information together with new synthetic views. The estimated depths can be used for convincing insertion of emoticons in the 3D space. This figure shows some cyclopean views generated by our algorithm where a rotating “emoticon” has been inserted. Notice that in the 3rd and 4th images the emoticon heart is half occluded by the neck of the person, thus demonstrating correct handling of occlusion events. The two left and right input sequences are not shown here.*

- Pattern Recognition*, pages 506–512, 1992.
- [2] C. Buehler, S. Gortler, M. Cohen, and L. McMillan. Min surfaces for stereo. In *Proc. Europ. Conf. Computer Vision*, Copenhagen, Denmark, May 2002.
 - [3] E. Chen and L. Williams. View interpolation for image synthesis. In *SIGGRAPH*, pages 279–288, 1993.
 - [4] I. Cox, M. Ott, and J.P. Lewis. Videoconference system using a virtual camera image. *US Patent*, 5,359,362, 1993.
 - [5] I. J. Cox, S. L. Hingorani, S. B. Rao, and B. M. Maggs. A maximum-likelihood stereo algorithm. *Computer Vision and Image Understanding*, 63(3):542–567, May 1996.
 - [6] A. Criminisi, J. Shotton, A. Blake, and P. Torr. Gaze manipulation for one-to-one teleconferencing. In *Proc. International Conference on Computer Vision*, Nice, Oct 2003.
 - [7] J. Gemmell, K. Toyama, C. Zitnick, T. Kang, and S. Seitz. Gaze awareness for videoconferencing: A software approach. *IEEE Multimedia*, 7(4), 2000.
 - [8] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521623049, 2000.
 - [9] H. Ishikawa and D. Geiger. Occlusions, discontinuities, and epipolar lines in stereo. In *European Conference on Computer Vision*, pages 232–248, Freiburg, Germany, 1998.
 - [10] V. Kolmogorov and R. Zabih. Computing visual correspondence with occlusions using graph cuts. In *International Conference on Computer Vision*, pages II:508–515, Vancouver, Canada, 2001.
 - [11] V. Kolmogorov and R. Zabih. Multi-camera scene reconstruction via graph cuts. In *Proc. Europ. Conf. Computer Vision*, pages 82–96, Copenhagen, Denmark, May 2002.
 - [12] Y. Ohta and T. Kanade. Stereo by intra- and inter- scanline search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7(2):139–154, 1985.
 - [13] S. Roy and I.J. Cox. A maximum-flow formulation of the n-camera stereo correspondence problem. In *Proc. Int. Conf. Computer Vision*, pages 492–499, 1998.

- [14] D. Scharstein. *View Synthesis Using Stereo Vision*, volume 1583 of *Lecture Notes in Computer Science (LNCS)*. Springer-Verlag, 1999.
- [15] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Int. J. Computer Vision*, 47(1–3):7–42, 2002.
- [16] J. Sun, H. Y. Shum, and N. N. Zheng. Stereo matching using belief propagation. In *Proc. Europ. Conf. Computer Vision*, Copenhagen, Denmark, May 2002.
- [17] R. Szeliski. Prediction error as a quality metric for motion and stereo. In *Proc. Int. Conf. on Computer Vision*, pages 781–788, Kerkyra, Greece, 1999.
- [18] T. Vetter. Synthesis of novel views from a single face image. *Int. J. Computer Vision*, 28(2):103–116, 1998.
- [19] R. Yang and Z. Zhang. Eye gaze correction with stereovision for video tele-conferencing. In *Proc. Europ. Conf. Computer Vision*, volume 2, pages 479–494, Copenhagen, Denmark, May 2002.