The SPS Algorithm: Patching Figural Continuity and Transparency by *Split-Patch Search*

A. Criminisi and A. Blake

Microsoft Research Ltd., 7 J J Thomson Ave, Cambridge, CB3 0FB, UK

antcrim@microsoft.com

Abstract

This paper describes a novel algorithm for the efficient synthesis of high-quality virtual views from only two input images. The emphasis is on the recovery of continuity of objects boundaries (figural continuity) with faithful synthesis of transparency effects.

The contribution of this paper is two-fold: i) the Split-Patch Search (SPS) technique is introduced for dense stereo which handles transparency effects by assigning multiple disparities to mixed pixels; ii) an efficient extension of exemplar-based image synthesis to the case of two-camera stereo is proposed. Furthermore, this paper presents an approximate but effective solution to the challenging problem of layer estimation and compositing in the case of small image patches.

The effectiveness of the proposed technique is demonstrated on a number of stereo image pairs taken from two-camera video-conferencing setups, where the quality of the synthesized talking heads is of paramount importance. Moreover, the improvement in the quality of image synthesis is quantified by comparing the output of the SPS algorithm with thirteen ground-truth images.

Figure 1: **High-quality two-camera virtual-view synthesis.** (a,b) Left and right input images (size 320×240) with large disparities and occlusions (60 pixel max disparity). (c) Virtual cyclopean image (detail) recovered by a standard dense stereo algorithm [7]. Along the boundary of the head streaky and blocky artefacts and aliasing effects occur. (c') Virtual cyclopean image (detail) after the proposed SPS enhancement step. The removal of artefacts and the introduction of mixed pixels produce a more natural-looking synthetic image.

1 Introduction

This paper deals with the problem of *efficiently* generating good-quality virtual images from stereo pairs. The example in fig. 1 is used throughout the paper to explain the steps of the proposed algorithm.

Many state of the art view-synthesis algorithms [3, 13, 15, 16] are prone to artefacts such as: i) aliasing and imperfect rendering of transparency effects, ii) streaky or blocky artefacts which disrupt figural continuity, iii) fattening or shrinking of foreground objects (see the corrupted outline of the head in fig. 1c). The goal of this paper is that of efficiently detecting and correcting those artefacts.

As observed in [18], mixed pixels occur along object boundaries of opaque objects and where there is transparency. In those situations, geometry-based techniques which assume a single depth per pixel, are inadequate. The problem is exacerbated in practical stereo matching when multi-pixel windows are used for correspondence matching. The window problem may be mitigated by the use of *split* or *shiftable* windows [12, 17], but proper modeling of transparency effects is also needed.

Our proposed approach for rendering can be seen as an extension of recent exemplar-based synthesis techniques [8, 11] to stereo. It is inspired by the work of Fitzgibbon *et al.* [9] who realised the potential of dictionaries of exemplars in procuring high quality detail at boundaries and over texture. Their virtual-view synthesis algorithm operates on a collection of *calibrated* input images (26 or more in their examples) to produce interpolated views of striking quality. They use a patch dictionary, gathered from the sequence it-

self, to define patch priors. One important property of this approach is that images are rendered without the need for explicit matting, simply by stealing pixels from appropriate locations in their rich dictionary. However their approach is very slow owing to the use of a substantial dictionary and a comprehensive but expensive treatment of the data likelihood. In contrast, our aim is to develop an effective strategy for artefacts which is nonetheless efficient enough to be included, on the fly, with real-time stereo matching.

The "SPS" — Split-Patch Search — algorithm achieves high computational efficiency (quasi real-time) without sacrificing image quality. Efficiency is achieved by a variety of means:

- restricting candidate patches to those lying on corresponding (left or right) epipolar lines;
- constraining the search region using tight, geometric depth bounds;
- applying exemplar-based synthesis *sparsely*, only where flagged by an inconsistency test.

Elsewhere, away from detected artefacts, synthesis is conventional and geometrically-based, and hence efficient. This parsimonious approach would fail however in the algorithm of Fitzgibbon et al. which relies on a plentiful supply of patches to achieve accurate rendering. Instead, repaired patches in SPS are composed of two part-patches, one attributed to the foreground and one to the background. This in turn demands the detection and representation of multiple depths, one for background and one for foreground, and is achieved by testing explicitly both foreground and background depth hypotheses. Transparency effects are rendered by effective compositing of the foreground and background portions to achieve realistic-looking virtual images.

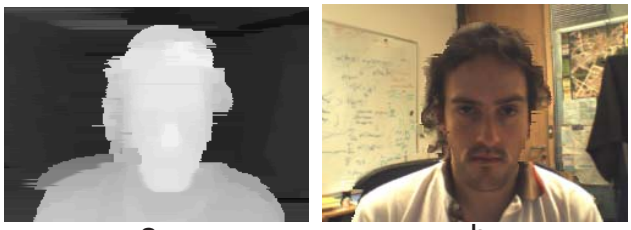
2 Problem Statement and Notation

This paper assumes given the two left and right input images I_l and I_r which have been epipolar-rectified (as opposed to the full camera calibration of [9]) and photometrically registered.

Our goal. We seek an efficient (possibly real-time) algorithm for the high-quality synthesis of the image that would be seen by a virtual camera placed in a new viewpoint. For simplicity of explanation, the focus here is on the synthesis of *cyclopean* images[1]. The extension to the case of general virtual viewpoint is straightforward.

Notation. Image points are indicated by boldface letters, *e.g.* $\mathbf{p}$ or $\mathbf{q}$. Uppercase typesets indicate matrices, *e.g.* $\mathtt{A}$. Capital letters indicate images or patches (subimages), *e.g.* I or Π. Furthermore, $\Pi_{\mathbf{p}}$ indicates a patch centred on the point $\mathbf{p}$ and $\Pi_{\mathbf{p}}(\mathbf{q})$ denotes the colour (or grey-scale intensity) of point $\mathbf{q}$ contained in the patch $\Pi_{\mathbf{p}}$. Finally, superscripts ^{f} and ^{b} indicate foreground and background patches, respectively.

[1]Cyclopean image (denoted I) is defined as the image that would be seen by a camera positioned half-way between the two input cameras.



a b

Figure 2: **Geometry-based virtual view synthesis.** (a) The disparity map D computed from the two input images in fig. 1a,b. The disparities are computed with reference to the cyclopean coordinate system. (b) The reconstructed virtual cyclopean image (denoted I in the text). The left and right occluded regions (around the foreground head) have been filled with pixels extracted from the left and right input views by making use of the fronto-parallel background assumption [7].

3 New-View Synthesis by SPS

This section outlines our view-synthesis algorithm which is composed of two steps. In the first step a standard dense-stereo technique generates a rough virtual image I. In the second phase, the image I is efficiently refined by the *Split-Patch Search* algorithm to produce the final virtual image I'. The main contribution of this paper is the *SPS* technique for efficient, exemplar-based image synthesis.

3.1. Estimating disparity and occlusion maps. Given the two input images I_l and I_r a disparity map D is generated with respect to the coordinate system defined by the desired virtual viewpoint and, at the same time, the virtual image I is synthesized (see fig. 2). For this purpose we use the algorithm in [7] but, as demonstrated in the results section, the refinement step of our algorithm is independent of the choice of the specific dense-stereo reconstruction technique. The main contribution of this paper lies in the way the unavoidable artefacts of I are removed. The next sections describe: i) an algorithm for the detection of the artefacts in I, and, ii) an algorithm for the removal of such artefacts by guided exemplar-based image re-synthesis.

3.2. Artefact detection and ordering. Given the input left and right images (I_l and I_r, respectively), the disparity map D and the corresponding occlusion map O (a product of dense stereo), each input image can be projected into the new, desired viewpoint. Let us call I_l^w the result of projecting the left input image I_l into the target viewpoint (see fig. 3a); and I_r^w for the right input image (fig. 3b). A pixel-wise distance $d(I_l^w, I_r^w)$ between the two projected images

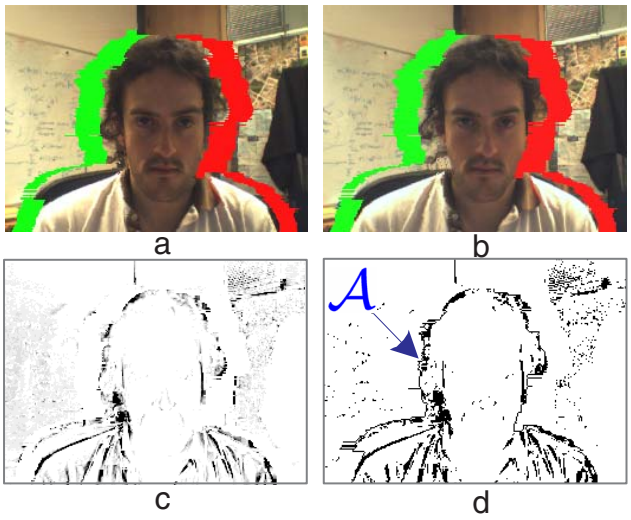


Figure 3: **Artefact detection.** (a,b) Projections of left and right input images into the target viewpoint, respectively (denoted I_l^w and I_r^w in text). Green and red regions denote estimated half-occlusions. (c) Aliasing-insensitive image distance map, $d(I_l^w, I_r^w)$. Darker points correspond to larger pixel distances (pixel intensities have been rescaled for improved visibility). (d) Detected set of artefact pixels $\mathcal{A}$ ($\approx 7\%$ of image area). These pixels will be re-synthesized and corrected by the SPS algorithm.

indicates the location and entity of artefacts (the dark points in fig. 3c). Assuming low levels of image noise, large values of $d(I_l^w, I_r^w)$ occur in places where the dense-stereo algorithm has failed to estimate the correct pixel correspondence between the two input images I_l and I_r. Note that inaccurate disparities do not necessarily produce inaccurate pixel synthesis; however, here, since we are interested in quality of image synthesis, we are correctly measuring artefacts in image space rather than disparity space. Furthermore, in order to overcome issues related to the (often) discrete nature of the disparity map, it is convenient to define the image distance $d(I_1, I_2)$ between two generic images I_1 and I_2 as a sampling-independent function [1], where half-occlusions are ignored.

Finally, artefacts are defined as the set $\mathcal{A}$ of points $\mathbf{p} \in I$ such that $d(I_l^w, I_r^w) > \lambda$ (fig. 3d) with λ a predefined value[2]. Furthermore, we have found it helpful to augment $\mathcal{A}$ with a one-pixel-wide boundary of the foreground. This can be achieved readily from the detected left and right occlusions. The algorithm then proceeds to the removal of the artefacts of the cyclopean image by a re-synthesis process.

As it will be clearer later this refinement procedure can be interpreted as an extension of the many exemplar-based texture and image synthesis algorithms [8, 11] to two-view stereo. The work of [2, 6, 10] has pointed out that exemplar-based synthesis benefits from processing the most reliable pixels first. Here we follow the same philosophy and assign a priority value $P(\mathbf{p})$ to each of the pixels $\mathbf{p} \in \mathcal{A}$ with the synthesis proceeding from highest- to lowest-priority pixels. Similar to [2] we adopt $P(\mathbf{p})$ to be proportional to the number of already filled neighbouring pixels although more sophisticated ordering algorithms may be employed [6, 10].

[2]Typically we choose λ very small (*e.g.* $\lambda = 5$ intensity levels) since the quality of image synthesis is fairly robust to large numbers of false positives.

3.3. Artefact removal by SPS and re-rendering. The algorithm proceeds as follows: We have given the first (corrupt) estimate of the cyclopean image I, and the set of detected artefacts $\mathcal{A}$. For each point $\mathbf{p} \in \mathcal{A}$ we extract the *source* patch $\Phi_\mathbf{p}$ (fig. 4, typically 5×5), centred on $\mathbf{p}$ and we seek a new, *target* patch $\Psi_\mathbf{p}$ which is similar to $\Phi_\mathbf{p}$ but where the artefacts have been removed (*cf.* fig. 6). Replacing $\Phi_\mathbf{p}$ with $\Psi_\mathbf{p}$ for all the points $\mathbf{p}$ in $\mathcal{A}$ achieves the desired correction. The steps of one iteration of the artefact-removal algorithm are:

- *Split-Patch Search*: given $\Phi_\mathbf{p}$ centred on $\mathbf{p}$, search along the corresponding scanlines in I_l and I_r for the two patches that are most similar to the foreground and background portions of $\Phi_\mathbf{p}$;
- *Compositing and Rendering*: combine those patches to generate the *target* patch $\Psi_\mathbf{p}$ and replace $\Phi_\mathbf{p}$ with $\Psi_\mathbf{p}$.

The details of each step are explained next.

3.3.1. Split-Patch Search. Given $\mathbf{p} \in \mathcal{A}$, its corresponding patch $\Phi_\mathbf{p}$ and a low-pass filtered version $\tilde{D}$ of the cyclopean disparity map D, we compute the foreground and background weight arrays $\Omega_\mathbf{p}^f$ and $\Omega_\mathbf{p}^b$ as follows:

$$\Omega_\mathbf{p}^f(\mathbf{q}) = \frac{\tilde{D}(\mathbf{q}) - \tilde{D}^{min}}{\tilde{D}^{max} - \tilde{D}^{min}}; \quad \Omega_\mathbf{p}^b(\mathbf{q}) = 1 - \Omega_\mathbf{p}^f(\mathbf{q}); \quad \forall \mathbf{q} \in \Phi_\mathbf{p} \tag{1}$$

with $\tilde{D}^{min}$ and $\tilde{D}^{max}$ respectively the minimum and maximum value of the (filtered) disparities within $\Phi_\mathbf{p}$. Notice that larger values of $\Omega_\mathbf{p}^f$ (*cf.* fig. 4) occur for points which are closer to the observer, and thus more likely to be foreground in a patch straddling foreground and background. $\Omega_\mathbf{p}^b$ is the complement of $\Omega_\mathbf{p}^f$.

These approximate foreground and background weights are sufficient to drive the search algorithm described below. The low-pass filtering of the disparities achieves robustness of the search algorithm by reducing the influence of high-frequency disparity artefacts (*e.g.* the horizontal streaks along the head boundary in fig. 2a). Moreover, robust variants of the weights in (1) may be defined, *e.g.* by means of an activation-like sigmoid transformation. In practice, though, we have found the definitions in (1) sufficient.

The SPS algorithm proceeds by searching for the two patches that are most similar to the foreground and background portions of the source cyclopean patch $\Phi_\mathbf{p}$. The recovered pair of left-view patches are denoted $L_\mathbf{p}^f$ and $L_\mathbf{p}^b$,

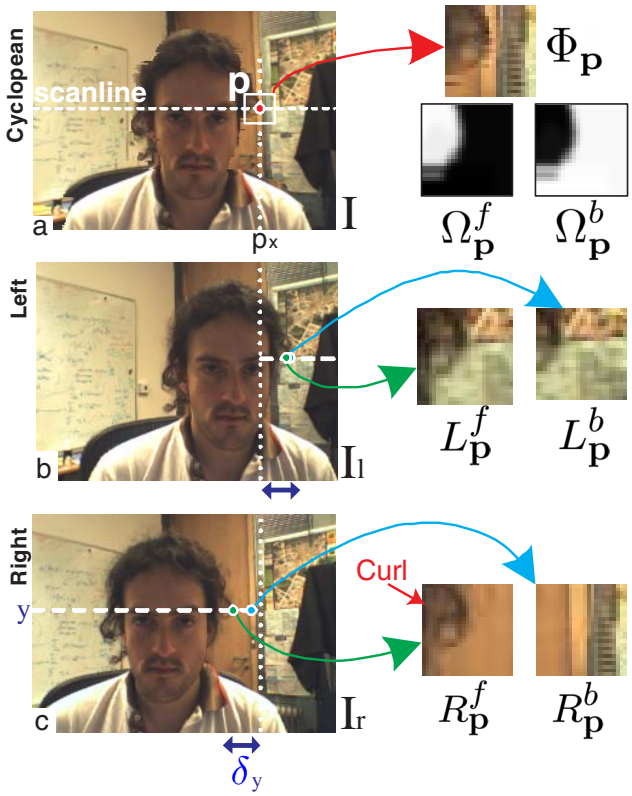


Figure 4: **Split-Patch Search.** Given the source patch $\Phi_{\mathbf{p}}$ in the corrupt cyclopean image I, we seek the two patches in I_l and I_r which are most similar to the foreground and background regions of $\Phi_{\mathbf{p}}$, respectively. Results of the automatic search are the pairs of patches $L^f_{\mathbf{p}}$ and $L^b_{\mathbf{p}}$ for the left input image and $R^f_{\mathbf{p}}$ and $R^b_{\mathbf{p}}$ for the right input image. The automatically computed value of δ_y determines the small portion of the current scanline in which the search is performed. In this running example we use patches of size 25×25 for clarity, however the SPS algorithm normally employs 5×5 patches for efficiency.

and the right-view ones $R^f_{\mathbf{p}}$ and $R^b_{\mathbf{p}}$ (fig. 4). It is important to stress that the search is limited, for efficiency, to small segments along the corresponding left and right scanlines as follows:

$$L^f_{\mathbf{p}} = L_{\hat{\mathbf{q}}} \text{ with } \hat{\mathbf{q}} = \arg \min_{p_x \le q_x \le p_x + \delta_y} d'(\Omega^f_{\mathbf{p}} * \Phi_{\mathbf{p}}, \Omega^f_{\mathbf{p}} * L_{\mathbf{q}})$$

$$L^b_{\mathbf{p}} = L_{\hat{\mathbf{q}}} \text{ with } \hat{\mathbf{q}} = \arg \min_{p_x \le q_x \le p_x + \delta_y} d'(\Omega^b_{\mathbf{p}} * \Phi_{\mathbf{p}}, \Omega^b_{\mathbf{p}} * L_{\mathbf{q}})$$

$$R^f_{\mathbf{p}} = R_{\hat{\mathbf{q}}} \text{ with } \hat{\mathbf{q}} = \arg \min_{p_x - \delta_y \le q_x \le p_x} d'(\Omega^f_{\mathbf{p}} * \Phi_{\mathbf{p}}, \Omega^f_{\mathbf{p}} * R_{\mathbf{q}})$$

$$R^b_{\mathbf{p}} = R_{\hat{\mathbf{q}}} \text{ with } \hat{\mathbf{q}} = \arg \min_{p_x - \delta_y \le q_x \le p_x} d'(\Omega^b_{\mathbf{p}} * \Phi_{\mathbf{p}}, \Omega^b_{\mathbf{p}} * R_{\mathbf{q}})$$

with $L_{\mathbf{q}}$ and $R_{\mathbf{q}}$ generic left-view and right-view patches centred on the generic point $\mathbf{q} \mid q_y = p_y$. Here the symbol $*$ denotes point-wise multiplication between images (or patches). The distance $d'(\Pi_1, \Pi_2)$ between two generic patches Π_1 and Π_2 is defined as the sum of squared distances (SSD) of pixel values where artefact pixels are ignored. The value δ_y which restricts the search region for

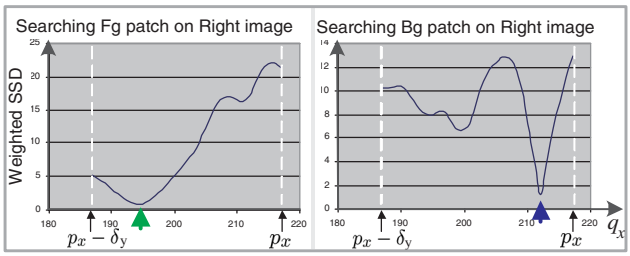


Figure 5: **Patch distances.** These two plots show the values of the weighted SSD distance functions $d'(\Omega^f_{\mathbf{p}} * \Phi_{\mathbf{p}}, \Omega^f_{\mathbf{p}} * R_{\mathbf{q}})$ and $d'(\Omega^b_{\mathbf{p}} * \Phi_{\mathbf{p}}, \Omega^b_{\mathbf{p}} * R_{\mathbf{q}})$ for varying values of $q_x \in [p_x - \delta_y, p_x]$ along the right scanline. $\Phi_{\mathbf{p}}$ is the example source patch in fig. 4. Restricting the search to a small scanline segment of length δ_y achieves efficiency. Our winner-take-all algorithm selects the patches $R^f_{\mathbf{p}}$ and $R^b_{\mathbf{p}}$ corresponding to the minima of the above plots (marked in green and blue, *cf.* fig. 4). These correspond to the two correct foreground and background disparities at point $\mathbf{p}$.

efficiency is a scanline-dependent value defined as $\delta_y = \max_{\mathbf{q} \in I \mid q_y = y} \tilde{D}(\mathbf{q})/2$. Examples of such automatically extracted patches are shown in fig. 4.

The SPS algorithm may be interpreted as a *winner-take-all* algorithm for dense stereo. However, unlike previous approaches of this kind, here the algorithm is applied twice, once to the foreground and once to the background portions of the source patch $\Phi_{\mathbf{p}}$. This has the effect of assigning *two* depth values to the artefact pixels in $\mathcal{A}$. Notably, in the case of mixed pixels the two estimated depths correspond to the depths of the foreground and background components of the mix. The reduced search region and the large autocorrelation of the $\Phi_{\mathbf{p}}$ patch[3] make the typically fragile winner-take-all algorithm sufficiently robust.

Efficiency. It is important to stress that the SPS algorithm is economical since for each point $\mathbf{p} \in \mathcal{A}$ the search region is restricted to a short scanline segment of length δ_y. Figure 5 shows the typical behaviour of the patch distance function for varying values of the q_x coordinate.

3.3.2. Selecting the best background patch. Figure 4 demonstrates the successful detection of the two left and right foreground patches $L^f_{\mathbf{p}}$ and $R^f_{\mathbf{p}}$ (the foreground hair curl is present in both and in the same position). However, due to occlusion, only one of the two retrieved background patches is meaningful. In fact, the true background of $\Phi_{\mathbf{p}}$ (the vertical door frame) is occluded in the left view I_l, thus the retrieved patch $L^b_{\mathbf{p}}$ is meaningless. In contrast, the right background patch $R^b_{\mathbf{p}}$ contains the correct background information. The actual choice between $L^b_{\mathbf{p}}$ and $R^b_{\mathbf{p}}$ is performed automatically by retaining the patch $\Pi^b_{\mathbf{p}}$ which is most similar to the background of $\Phi_{\mathbf{p}}$, *i.e.* : $\Pi^b_{\mathbf{p}} = \arg\min_{\Lambda \in \{L^b_{\mathbf{p}}, R^b_{\mathbf{p}}\}} d'(\Omega^b_{\mathbf{p}} * \Lambda, \Omega^b_{\mathbf{p}} * \Phi_{\mathbf{p}})$. In the example in fig. 4 the selected background patch is $\Pi^b_{\mathbf{p}} = R^b_{\mathbf{p}}$.

[3] Artefacts tipically occur along high-contrast object boundaries.

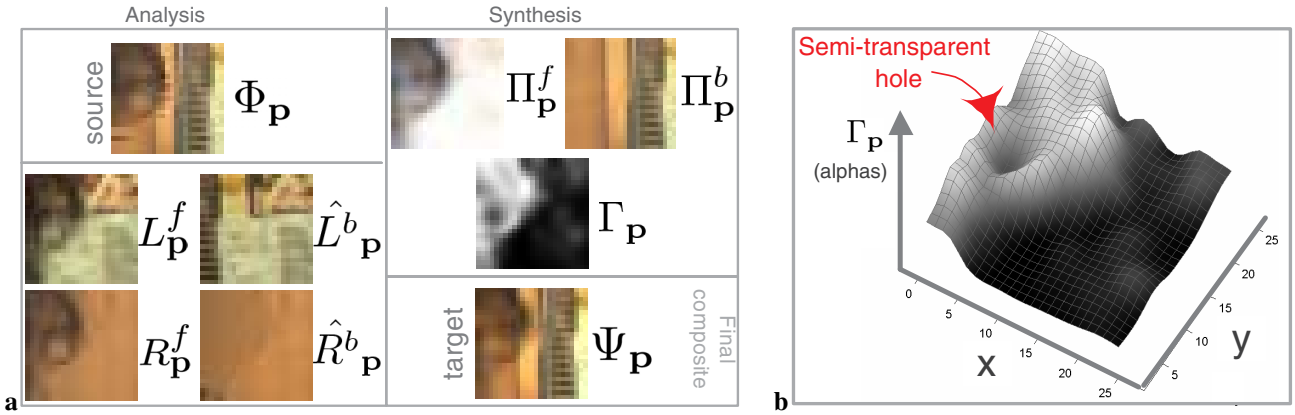


Figure 6: **Patch matting and compositing.** See text for notation. (a) Once the two backgrounds ($\hat{L}^b{}_{\mathbf{p}}$,$\hat{R}^b{}_{\mathbf{p}}$) associated to the two foreground patches ($L^f_{\mathbf{p}}$,$R^f_{\mathbf{p}}$) have been estimated transparencies ($\Gamma_{\mathbf{p}}$) and foreground colours ($\Pi^f_{\mathbf{p}}$) of the target patch can be computed. Finally, the target patch $\Psi_{\mathbf{p}}$ is computed by the conventional compositing equation (2). The patch $\Psi_{\mathbf{p}}$ is a clear improvement with respect to the original, corrupted cyclopean patch $\Phi_{\mathbf{p}}$. (b) A 3D height-map visualization of the estimated opacities $\Gamma_{\mathbf{p}}$ showing the correctly recovered semi-transparent hole in the hair curl.

At this point we have computed $\Pi^b_{\mathbf{p}}$ which is one of the elements needed for compositing the artefact-free target patch $\Psi_{\mathbf{p}}$ (fig. 6). Further steps are: i) for each pixel $\mathbf{q} \in \Phi_{\mathbf{p}}$ estimating its (uncontaminated) foreground colour $\Pi^f_{\mathbf{p}}(\mathbf{q})$ and transparency $\Gamma_{\mathbf{p}}(\mathbf{q})$. ii) combine foreground $\Pi^f_{\mathbf{p}}$, background $\Pi^b_{\mathbf{p}}$ and transparencies $\Gamma_{\mathbf{p}}$ to obtain the desired target patch $\Psi_{\mathbf{p}}$. These steps are described next.

3.3.3. Patch matting, compositing and rendering. For each point $\mathbf{p} \in \mathcal{A}$, we have described how to extract two foreground-registered patches $L^f_{\mathbf{p}}$ and $R^f_{\mathbf{p}}$. If we knew their corresponding backgrounds we could apply the technique described in [19] to estimate pixel opacities and uncontaminated foreground colours necessary to generate the target patch $\Psi_{\mathbf{p}}$. This section attacks this problem; however, having available only *two* input images makes the problem ill-posed, and reasonable assumptions will be necessary. Segmentation-based matting techniques [4] are not suited here since they target *single*-image cases. The additional information provided by the comparison of the *two* foreground patches is exploited in this paper.

We begin by noting that the patch $L^f_{\mathbf{p}}$ extracted from the *left* input image can be interpreted itself as a composite image. In our example, its background $\hat{L}^b{}_{\mathbf{p}}$ (the poster on the back wall) is completely visible in the *right* input view and can be extracted by the following search process:

$$\hat{L}^b{}_{\mathbf{p}} = R_{\hat{\mathbf{q}}} \text{ where } \hat{\mathbf{q}} = \arg \min_{p_x \le q_x \le p_x + \delta_y} d'(\Omega^b_{\mathbf{p}} * L^f_{\mathbf{p}}, \Omega^b_{\mathbf{p}} * R_{\mathbf{q}})$$

and simmetrically for $\hat{R}^b{}_{\mathbf{p}}$. In our running example, however, the background corresponding to the right foreground patch $R^f_{\mathbf{p}}$ (the brown door in fig. 4c) is occluded, in the left image, by the person. Thus, the automatically extracted patch $\hat{R}^b{}_{\mathbf{p}}$ is meaningless. This situation can be automatically detected by checking the sign of the quantity Δ defined as:

$$\Delta = d'(\Omega^b_{\mathbf{p}} * \hat{R}^b{}_{\mathbf{p}}, \Omega^b_{\mathbf{p}} * R^f_{\mathbf{p}}) - d'(\Omega^b_{\mathbf{p}} * \hat{L}^b{}_{\mathbf{p}}, \Omega^b_{\mathbf{p}} * L^f_{\mathbf{p}}).$$

In fact, the situation presented in our example corresponds to a value $\Delta > 0$. Similarly, $\Delta < 0$ corresponds to the occlusion of the left background patch $\hat{L}^b{}_{\mathbf{p}}$.

The problem of occluded background patches is of a general nature and assumptions are needed to estimate the missing information. In the case of occluded $\hat{R}^b{}_{\mathbf{p}}$ we proceed as follows: given the right foreground patch $R^f_{\mathbf{p}}$ and the background filter $\Omega^b_{\mathbf{p}}$, we extract the pixels of $R^f_{\mathbf{p}}$ which belong to the background and then fit a parametric surface model (*e.g.* polynomial, spline etc.) to the corresponding colour values[4]. Finally, the fitted surface model is used to extrapolate the colours of the pixels in the occluded portion of $R^f_{\mathbf{p}}$. We have found that for small patches (5×5) extrapolation via a generic planar fit (generally not at constant height) produces good results. More powerful extrapolation techniques may be considered for larger and highly textured patches. Figure 6 shows the estimated $\hat{R}^b{}_{\mathbf{p}}$ patch of the example; notice the extrapolated area behind the hair curl. Symmetrical reasoning applies when $\Delta < 0$.

Now, similarly to [19] we have available two known foreground-registered patches ($L^f_{\mathbf{p}}$ and $R^f_{\mathbf{p}}$) and the two corresponding (different) background patches ($\hat{L}^b{}_{\mathbf{p}}$ and $\hat{R}^b{}_{\mathbf{p}}$). For $L^f_{\mathbf{p}}$ and $R^f_{\mathbf{p}}$ the conventional compositing equation generalized to the case of patches is:

$$L^f_{\mathbf{p}} = \Gamma_{\mathbf{p}} * \Pi^f_{\mathbf{p}} + (1 - \Gamma_{\mathbf{p}}) * \hat{L}^b{}_{\mathbf{p}}$$
$$R^f_{\mathbf{p}} = \Gamma_{\mathbf{p}} * \Pi^f_{\mathbf{p}} + (1 - \Gamma_{\mathbf{p}}) * \hat{R}^b{}_{\mathbf{p}}$$

[4] we employ an RGB colour model.

with $\Gamma_{\mathbf{p}}$ the opacities and $\Pi^f_{\mathbf{p}}$ the uncontaminated foreground colours. Since both background patches are known, then both $\Gamma_{\mathbf{p}}$ and $\Pi^f_{\mathbf{p}}$ are uniquely determined. Opacities are assumed to apply equally to each of the RGB channels.

Unfortunately, some of the corresponding pixels in the two backgrounds $\hat{L^b}_{\mathbf{p}}$ and $\hat{R^b}_{\mathbf{p}}$ may have very similar colours, thus making the accurate recovery of transparencies and foreground colours ill-posed [19]. Image noise can further exacerbate this pathological situation. However, reasonable estimates of transparencies and colours can be obtained through the incorporation of prior information (*e.g.* on the distribution of alpha and colour values). This regularization effect can be achieved either by means of a Bayesian approach [19] or, simply by a depth-driven, low-pass filtering of the transparency and colour signals. We have found the latter to work sufficiently well in the case of small image patches. Examples of estimation of $\Gamma_{\mathbf{p}}$ and $\Pi^f_{\mathbf{p}}$ are shown in fig. 6, where foreground colours have been composited on a white background for aided visualization.

Finally, given the foreground $\Pi^f_{\mathbf{p}}$, the opacities $\Gamma_{\mathbf{p}}$ and the background $\Pi^b_{\mathbf{p}}$, the target patch $\Psi_{\mathbf{p}}$ remains defined:

$$\Psi_{\mathbf{p}} = \Gamma_{\mathbf{p}} * \Pi^f_{\mathbf{p}} + (1 - \Gamma_{\mathbf{p}}) * \Pi^b_{\mathbf{p}}. \qquad (2)$$

Figure 6 shows the results of running one iteration of the SPS algorithm on a real-image example: comparison between the original patch $\Phi_{\mathbf{p}}$ and the estimated target patch $\Psi_{\mathbf{p}}$, demonstrates the effective removal of artefacts. In fig. 6b notice how the estimated transparency map $\Gamma_{\mathbf{p}}$ correctly captures the semi-transparent nature of the hole in the hair curl. The enhancement of the entire virtual image I is achieved by copying the content of $\Psi_{\mathbf{p}}$ inside $\Phi_{\mathbf{p}}$ for all pixels $\mathbf{p} \in \mathcal{A} \cap \Phi_{\mathbf{p}}$ and repeating the steps above until all the pixels in $\mathcal{A}$ have been re-synthesized. Figure 1c' shows the result of applying the SPS enhancement algorithm to the entire cyclopean image in fig. 1c. In the current version of the algorithm a pixel $\mathbf{p} \in \mathcal{A}$ may be synthesized more than once since it belongs to a number of overlapping patches. In this case only the last value is retained. Moreover, we have found larger patch sizes to yield better quality of synthesis at the expense of CPU cycles. The SPS algorithm is validated next on a number of ground-truth data and further real-image examples.

4 Results and Comparisons

This section presents a quantitative and qualitative evaluation of the performance of the proposed SPS algorithm. The improvement in the quality of the synthetic image is measured by comparisons against ground-truth data. Further examples of virtual-image synthesis in typical two-camera video-conferencing sessions are also presented.

Comparison with ground truth. The performance of the SPS algorithm is measured as follows: given the input images I_l and I_r and the corresponding ground-truth cyclopean image I_{gt} we first synthesize the cyclopean view I using a standard dense-stereo algorithm. Secondly, we apply the outline-enhancement SPS algorithm to the image I and generate the improved image I'. The proportionate quality improvement is measured as the ratio

$$\rho = \frac{d(I, I_{gt}) - d(I', I_{gt})}{d(I, I_{gt})}.$$

Thus, positive values of ρ indicate an actual improvement of the image quality (because $d(I', I_{gt}) < d(I, I_{gt})$) and vice-versa for negative values of ρ. The image distance $d(I_1, I_2)$ between two generic images I_1 and I_2 is defined as the sampling-independent distance of [1]. These distances are computed only at the points labelled as artefacts.

Figure 7: **Additional ground-truth data.** Sample frames from the two additional ground-truth sequences used to quantify the performance of the SPS algorithm. The camera is translating horizontally with constant velocity. (a) `Oranges` sequence. Image size is 256×192 and max object occlusion is about 4% of the image width. (b) `Cube` sequence. Image size is 192×80 and max object occlusion is about 3% of the image width.

We run our experiments on *all* nine ground-truth sequences from the `Middlebury` data set (`www.middlebury.edu/stereo`). For each sequence we chose the first and last frames (frames 0 and 8) as the left and right input images and the middle frame (frame 4) as the ground-truth cyclopean image I_{gt}. The results are as follows:

data	`barn1`	`barn2`	`bull`
ρ	+0.20%	+1.57%	+7.33%
data	`cones`	`poster`	`sawtooth`
ρ	+2.94%	+6.59%	+2.33%
data	`teddy`	`tsukuba`	`venus`
ρ	+3.89%	+0.8%	+1.66%

It can be observed that in all the above experiments the sign of ρ is positive, confirming the actual improvement of image quality achieved by the SPS algorithm.

The `Middlebury` dataset uses short baselines and, consequently, is characterized by very small occluded regions. This is not very representative of real-world stereo pairs. For instance, the stereo images in fig. 1 are characterized by a maximum cyclopean occlusion of 8% of the image width (to be compared to the 2.6% average of maximum cyclopean occlusions in the `Middlebury` dataset). Thus, in order to test the SPS algorithm with larger occlusions we generated our own ground-truth data by acquiring two sequences from a horizontally-translating camera (fig. 7). From each of the two sequences we extracted two triplets of "left-cyclopean-right" images (labelled as "exp1" and "exp2"), and the measured values of ρ are as follows:

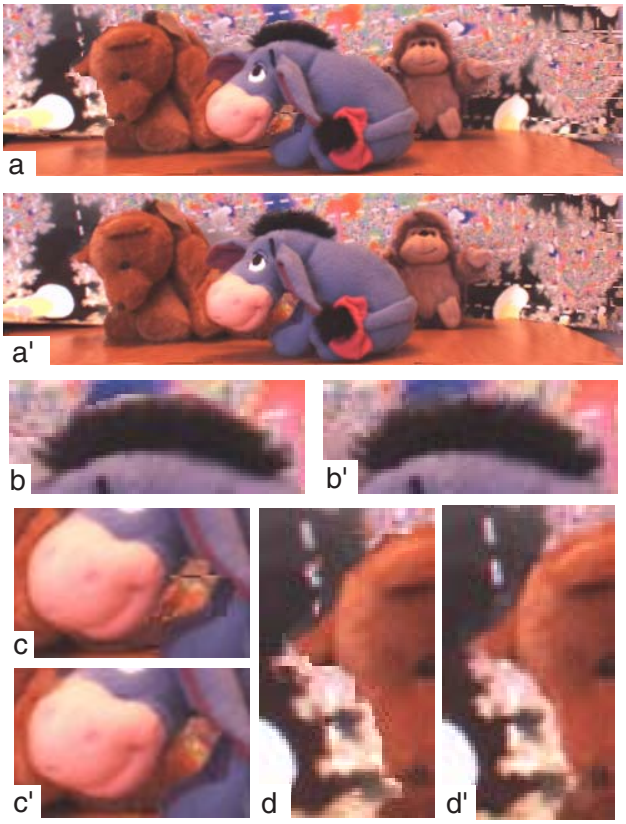


Figure 8: **A furry toy example.** (a) The cyclopean image generated by a standard geometry-based dense stereo algorithm. The input left and right images are not shown here. (a') The cyclopean image after SPS enhancement. (b,c,d) details of (a). (b',c',d') corresponding details of (a'). The artefacts in the donkey's hair, the donkey's neck and the teddy's nose have been removed and the quality of the synthetic images enhanced.

data	`oranges, exp1`	`oranges, exp2`
ρ	+6.50%	+3.52%
data	`cube, exp1`	`cube, exp2`
ρ	+5.47%	+9.95%

Once again, all the entries of the above table show positive values of ρ, thus confirming the efficacy of the SPS technique. The larger (on average) values of ρ in this second set of experiments are explained by the fact that larger occlusion regions are more likely to cause failure of the geometry-based synthesis. Consequently, the contribution of our exemplar-based enhancement becomes more evident.

A furry toy example. Figure 8 shows the results of synthesizing cyclopean images on a difficult furry toy example. The detail images highlight the removal of boundary blocks and streaks, and the improved rendering of mixed pixels.

Using different dense-stereo algorithms. To demonstrate the general nature of SPS we have applied it to the virtual images generated by different dense-stereo algorithms. Figures 9a,b show the cyclopean images obtained from the disparities estimated by the algorithms in [5] and [14], respectively. Figures 9a',b' show the corresponding SPS-enhanced cyclopean images.

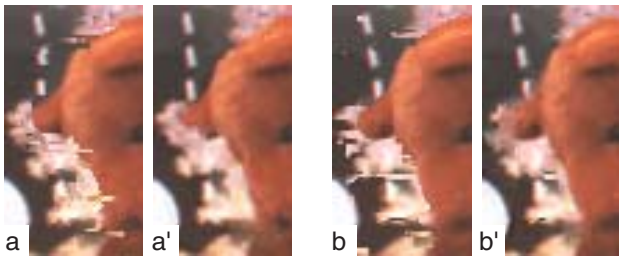


Figure 9: **SPS with different stereo algorithms.** (a,b) Virtual images generated by the Dynamic Programming algorithm in [5] and the Graph-Cut algorithm in [14], respectively. (a',b') The corresponding images after SPS enhancement. In both cases artefacts have been removed and the outline of the teddy bear enhanced.

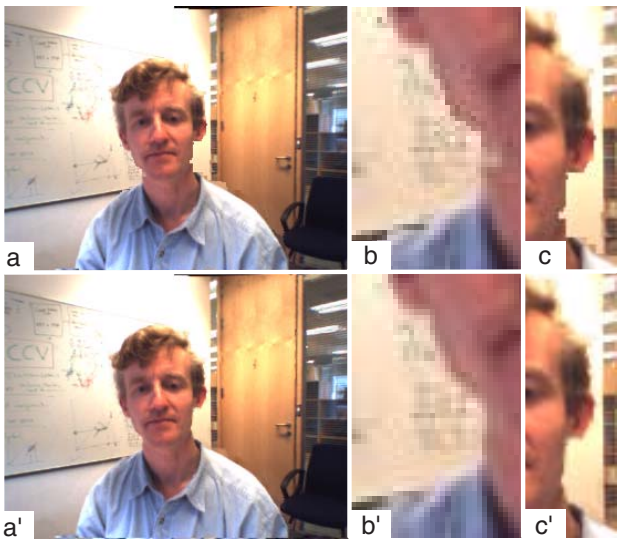


Figure 10: **A two-camera video-conferencing example.** (a) Cyclopean view after first pass. The two input images are not shown here. (b,c) Details of (a) showing aliasing artefacts along the head boundary. (a') Cyclopean view after SPS enhancement. (b',c') Details of (a') where the introduction of pixel mixing produces an enhanced outline of the foreground object.

Video-conferencing examples. Finally, we present two more video-conferencing examples (of the kind in fig. 1). Comparing fig. 10b' with fig. 10b and fig. 10c' with fig. 10c highlights the enhanced quality.

Figure 11 demonstrates that the SPS algorithm is particularly useful for background substitution. Notably, the unavoidable aliasing that arises from disparity-based background removal and substitution is fixed by running the SPS algorithm along the boundary of the foreground object, thus producing a smooth and artefact-free foreground/background transition.

Image sequences. Our experiments demonstrate that synthesized temporal sequences also benefit from the SPS algorithm, however, due to space constraints we are unable to

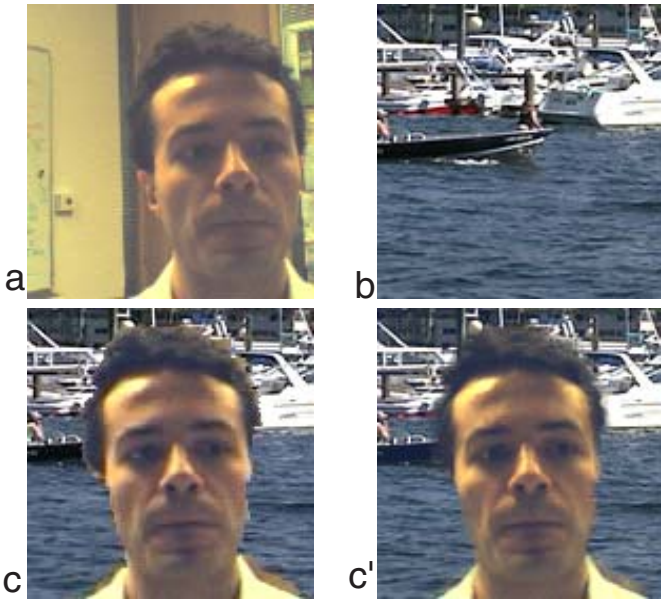


Figure 11: **SPS enhances background substitution.** (a) Input left image, the right image is not shown here. (b) Desired new background. (c) Replacing the desired background in the synthetic *cyclopean* view by simple depth thresholding produces unnaturally sharp foreground/background transitions. Moreover, lack of pixel mixing make the head appear "stuck" upon the background. (c') The result of applying SPS to (c). Boundary artefacts (aliasing and lack of pixel mixing) have been fixed, thus yielding a more realistic-looking composition.

show results here. Numerous examples and results on both static and temporal data are available at [20].

5 Conclusion

This paper has presented a novel technique for the efficient and accurate synthesis of virtual views from only two input images. The emphasis is on the synthesis of artefact-free objects boundaries (figural continuity) with faithful pixel mixing. We employ a two-pass new-view synthesis algorithm which combines the *efficiency* of disparity-based techniques with the *quality* of exemplar-based synthesis algorithms.

The main contribution is the *Split-Patch Search* algorithm for virtual-image synthesis which: i) detects the artefacts generated by the geometry-based synthesis and ii) removes them by means of a multiple-depth stereo algorithm. The actual image synthesis is performed in an exemplar-based fashion, adapted to the case of stereo images. Transparency effects and mixed pixels are rendered by patch-based matting and compositing.

Computational efficiency is achieved by taking advantage of the geometry-based reconstruction to constrain tightly the search for exemplar patches in the SPS step. In our C++ implementation, which exploits SSE2 instructions, the first pass (dense stereo) runs at approximately 7.5 f.p.s. on a dual-processor 3GHz Pentium IV with 1Gb RAM. The SPS refinement phase reduces the speed down to about 5.5 f.p.s. for artefacts which typically cover less than 10% of the image area. The result is a nearly real-time algorithm for the synthesis of high-quality virtual views from only two input images.

Areas of further research include: i) integrating SPS into stereo matching, ii) investigating a probabilistic framework for accurate patch matting in the challenging two-image scenario, iii) extending the SPS algorithm to explicitly impose temporal consistency in image sequences.

Acknowledgements. The authors would like to thank C. Rother and R. Szeliski for inspiring discussions, and G. Cross for the efficient implementation of our dense stereo and SPS algorithms.

References

[1] S. Birchfield and C. Tomasi. A pixel dissimilarity measure that is insensitive to image sampling. *IEEE Transactions on Pattern Analysis and Machine Intelligence,*, 4(20):401–406, 1998.

[2] R. Bornard, E. Lecan, L. Laborelli, and J.-H. Chenot. Missing data correction in still images and image sequences. In *ACM Multimedia*, France, December 2002.

[3] A. Broadhurst and R. Cipolla. A statistical consistency check for the space carving algorithm. In *Proc. ICCV*, 2001.

[4] Y.-Y. Chuang, B. Curless, D. H. Salesin, and R. Szeliski. A Bayesian approach to digital matting. In *Proc. CVPR*, 2001.

[5] I. J. Cox, S. L. Hingorani, S. B. Rao, and B. M. Maggs. A maximum-likelihood stereo algorithm. *CVIU*, 63(3):542–567, May 1996.

[6] A. Criminisi, P. Perez, and K. Toyama. Object removal by exemplar-based inpainting. In *Proc. CVPR*, Madison, WI, Jun 2003.

[7] A. Criminisi, J. Shotton, A. Blake, and P.H.S. Torr. Gaze manipulation for one-to-one teleconferencing. In *Proc. ICCV*, Nice, Oct 2003.

[8] A. Efros and T. Leung. Texture synthesis by non-parametric sampling. In *Proc. ICCV*, pages 1039–1046, Sep 1999.

[9] A. Fitzgibbon, Y. Wexler, and A. Zisserman. Image-based rendering using image-based priors. In *Proc. ICCV*, Nice, Oct 2003.

[10] P. Harrison. A non-hierarchical procedure for re-synthesis of complex texture. In *Proc. Int. Conf. Central Europe Comp. Graphics, Vis. and Comp. Vision*, Plzen, Czech Republic, February 2001.

[11] A. Hertzman, C. E. Jacobs, N. Oliver, B. Curless, and D. H. Salesin. Image analogies. In *Proc. ACM SIGGRAPH*, Eugene Fiume, 2001.

[12] H. Hirschmueller. Improvements in real-time correlation-based stereo vision. *IJCV*, 2002.

[13] R. Koch. 3D surface reconstruction from stereoscopic image sequences. In *Proc. ICCV*, pages 109–114, 1995.

[14] V. Kolmogorov and R. Zabih. Computing visual correspondence with occlusions using graph cuts. In *Proc. ICCV*, pages 508–515, 2001.

[15] K. N. Kutulakos and S. M. Seitz. A theory of shape by space carving. Technical Report CSTR 692, University of Rochester, 1998.

[16] D. Scharstein. *View Synthesis Using Stereo Vision*, volume 1583 of *Lecture Notes in Computer Science (LNCS)*. Springer-Verlag, 1999.

[17] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *IJCV*, 47(1/2/3):7–42, 2002.

[18] R. Szeliski and P. Golland. Stereo matching with transparency and matting. *IJCV*, 1998.

[19] Y. Wexler, A. Fitzgibbon, and A. Zisserman. Bayesian estimation of layers from multiple images. In *Proc. ECCV*, Copenhagen, 2002.

[20] `www.research.microsoft.com/vision/cambridge/i2i`