Scene estimation from a swiped image

Michael Lawson, *Member, IEEE,* Mike Brookes, *Member, IEEE,* and Pier Luigi Dragotti, *Fellow, IEEE,*

***Abstract*—The image blurring that results from moving a camera with the shutter open is normally regarded as undesirable. However, the blurring of the images encapsulates information which can be extracted to recover the light rays present within the scene. Given the correct recovery of the light rays that resulted in a blurred image, it is possible to reconstruct images of the scene from different camera locations. Therefore, rather than resharpening an image with motion blur, the goal of this paper is to recover the information needed to resynthesise images of the scene from different viewpoints. Estimation of the light rays within a scene is achieved by using a layer-based model to represent objects in the scene as layers, and by using an extended level set method to segment the blurred image into planes at different depths. The algorithm described in this paper has been evaluated on real and synthetic images to produce an estimate of the underlying Epipolar Plane Image.**

***Index Terms*—Plenoptic function, Plenoptic camera, Layer based depth, Blurred images**

I. Introduction

The image blurring that results from moving a camera whilst the shutter is open is normally regarded as a problem. However, the structure of the blurring within the image encapsulates information which can be extracted to recreate a view of the scene from an arbitrary camera location. We define a "swiped image" to be one in which objects are blurred due to the camera moving whilst the shutter remains open. A swiped image, shown in Figure 1(b), is created from the integration of the light rays present in the scene as the camera moves. Camera motion blur within the swiped image is due to the combination of the many light rays arriving at different viewpoints. Rather than merely resharpening the image, the goal of this paper is to retrieve the information needed to render a sharp images at any of these viewpoints. If the light rays that created the swiped image can be determined, then new views of the scene can be synthesised by creating a virtual camera at an arbitrary position.

A useful framework to describe how images of a scene change with the position of a camera is the plenoptic function. This was first introduced by Adelson and Bergen in [3], where a scene was described in terms of the light rays that travel from the surface of objects within the scene to a camera [3]–[8]. Obtaining the plenoptic function of a scene is normally not possible with a typical consumer device containing a single-lens camera. Plenoptic cameras, or arrays of cameras, can be used [9]–[12], but these are too complex and expensive for widespread adoption. The EPI (Epipolar Plane Image) [13], which is used in this paper, is a subset of the plenoptic function and is obtained when the camera is constrained to travel along a horizontal straight line and the scene is assumed to be static. An example of the images from which the EPI can be constructed is shown in Figure 1(a) and the resultant swiped image is shown in Figure 1(b).

Fig. 1. Integrating the light rays within an EPI of a scene, shown by (a), leads to a swiped image, shown by (b). The aim is to recover the EPI from a swiped image of a realistic scene.

Inferring depth in a scene from swiped images has previously been studied in [14], [15], where depth is inferred by estimating the degree of blurring that is present at each pixel within the image. However the image is deblurred from only a single perspective and without taking into account occlusions of objects in the scene caused by camera movement. The algorithm that is presented in the current paper extends this by recovering the EPI of the scene. This is achieved by modelling the scene as comprising layers at different depths, which has been shown to be a good approximation of real life scenes [16]–[19].

The challenges of recovering the EPI include that of estimating the depths present within the image, finding the silhouette of objects at these depths, recovering the texture surfaces of these objects, and finally rendering new images of the scene using the recovered objects and scene geometry. The task of estimating the depths present within the image is addressed by creating a histogram of estimated depths within the image and selecting histogram peak locations as the depths present within the scene. In this paper an extension to the level set method whereby the outline of depth layers can be extracted is demonstrated. The outline of the depth layers is then be used to recover the surface intensities of the planes. Once the boundaries of the objects within the scene have been determined, the scene texture is determined, and a new EPI can be created.

Michael Lawson is with the Department of Electrical and Electronic Engineering, Imperial College London, UK e-mail: (M.Lawson14@imperial.ac.uk). Part of this work was previously presented in [1], [2].

Besides introducing a variation of the level set method to segment blurred images (Section IV-B), a novel aspect of the algorithm presented here is also that, given the estimated depths within the scene, we create a forward model of the swiped image acquisition that explicitly models the occlusion of each layer by others. We use the knowledge of this forward model to recover the EPI. We then produce an estimate of the error in the recovered EPI by the colour separation of the layers, and iteratively improve the estimate of the forward model to recover the EPI (Section IV-D).

The technique presented assumes a highly controlled image acquisition, where the swiped image is obtained using a motorised camera slider. This is the ideal case for camera movement, reducing the number of free parameters that need to be estimated for the EPI to be recovered. A key contribution is a more accurate forward model of the acquisition of the swiped image, using knowledge of the scene geometry and the movement of the camera. Although this is not something that can be obtained by using a casual hand movement with a typical consumer device, the forward model of the acquisition could be further developed by adapting it to include knowledge of non-uniform camera movement. Methods of estimating camera movement already exists using the inertial sensors and accelerometers in smartphones, and are already used for high precision inputs to games [20].

The outline of this paper is as follows: Section II summarizes related work in this area. Section III presents an overview of the proposed algorithm. Section IV describes in detail the proposed method that allows the EPI to be calculated from the swiped image. Section V demonstrates the results of the proposed algorithm on real images. Finally, we conclude in Section VI.

II. Related work

The recovery of the EPI from a swiped image is related to that of recovering the plenoptic function, and to also the estimation of depth from motion. We briefly review both of these in the following subsections.

A. Plenoptic function recovery

Recovery of the plenoptic function of a scene is a problem that has been well studied in recent years. The full plenoptic function is complicated to analyse due to its high dimensionality, so much of the work has concentrated on simplified versions. One simplification of the full plenoptic function is the light field model [21], in which time and wavelength are fixed. The scene is assumed to lie within a bounding box, restricting the light field plenoptic function to 4 parameters $I(u, v, s, t)$. Each ray of light is defined by its intersections with two parallel planes. The plane with the coordinates (u, v) is the focal plane, and the plane with the coordinates (s, t) is the camera plane.

The recovery of the light field of a scene has been accomplished by using a grid of images of the scene [10], [12]. A grid of images may be obtained either by a dedicated plenoptic camera or by an array of cameras, taking multiple pictures of the scene. A further technique has been developed to recover the light field from a focal stack, in which the light field of a scene can be estimated from a set of images taken with different focuses [22]. This enables, for example, the refocusing of an image after it has been taken.

A further simplification to the plenoptic function can be made by constraining the camera movement to a horizontal line; this leads to the Epipolar Plane Image (EPI) [13]. In the EPI model, wavelength and time are again omitted, resulting in a 3-parameter function, $I(u, v, t)$.

The recovery of the entire EPI from a set of images taken along a line has been studied in, for example, [4], [5], [7]. The spectral properties of the plenoptic function have been characterised and the minimum sampling density of the EPI to create images from novel viewpoints without aliasing [4] has been established. Specifically, it has been shown that real images of complex scenes can be rendered using a layer-based model of the scene in which object depths are quantized to discrete values [16]–[19]. The advantages of the layer-based model are that it makes occlusion ordering explicit and allows the spectral properties of the scene to be well defined.

B. Depth from motion blur

Recovery of the EPI from a swiped image is also related to the problem of recovering depth from motion blur. Without the multiple images available in the cases discussed in Section II-A, the locations of objects within the swiped image must be estimated from the blur. Blur within an image may vary spatially, owing to variations in, for example, defocus blur or motion blur. Accurate estimation of blur for each region of an image can be used to create a sharp image of the scene from a single perspective.

Over the past two decades, a number of studies have addressed the issue of analysing motion blur, e.g. [23]–[26]. Frequently the motion blur is assumed to arise only from the movement of objects within a scene, whilst the camera remains stationary. The approach in [26] uses the statistics of the distribution of gradients within an image to estimate blur. Motion blur is modelled as 1D filtering with a box function along the direction of the movement of the subject in the scene during the exposure period. Each window within the image is artificially blurred in the other dimension to find the distribution of gradients that best matches the distribution of gradients in the direction of the motion blur. This technique can work well, but is dependent upon the assumption that similar blur in horizontal and vertical directions will produce similar gradient distributions, which may not be the case in image regions containing vertical or horizontal edges.

An alternative method of analysing spatially-variant motion blur is used in [23], with the shape of the Fourier transform of the pixels in a window being used to calculate the maximum likelihood blur kernel at each point.

III. Algorithm Overview

A. Problem setup

To gain insights into the proposed algorithm, we first consider the idealised setting shown in Figure 2. Our goal is to retrieve the EPI of a scene from a swiped image. We achieve

this by adopting a layer-based model in which we assume that a complex real-world scene is modelled as comprising P fronto-parallel planes. Each plane has a surface intensity corresponding to that of the real-life scene and the depth, or distance from the camera, of plane p is denoted as z_p. This is illustrated from above in Figure 2, for the case of $P = 3$. For this scene, we assume that there is a "background plane" that fills the field of view of the camera during the entire swipe, to ensure that the swiped image consists only of light rays from planes. The background plane is always the plane furthest from the camera, and has depth z_P. Using planes as depth layers is convenient for rendering and occlusion ordering.

The problem of recovering the EPI from a swiped image is very challenging, with a large number of unknowns during the capture of the image. If the swiped image was acquired during an arbitrary camera movement, we would simultaneously need to estimate both the path of the camera and the EPI. In this paper, we reduce the number of free variables by assuming that the camera is swiped from x_{01} to x_{02} at a constant velocity along a horizontal line parallel to the planes. This defines an additional plane, the zeroth plane, at a depth of $z_0 = 0$. This plane can also be viewed as a masking plane, as the situation equivalent to the camera swiping from minus infinity to plus infinity with the shutter open, but only being able to see the scene through the zeroth plane. We define the disparity, l_p, of each plane to be the shift in its location in the image as the camera moves between x_{01} and x_{02}. Given the swiped image $I_I(u, v)$, and knowledge of x_{01} and x_{02}, we wish to recover the EPI of the scene.

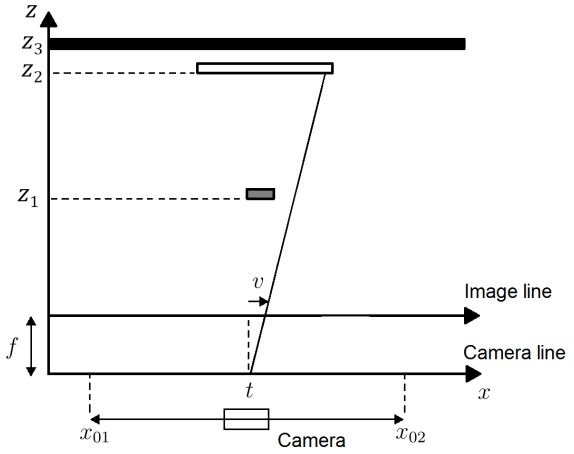


Fig. 2. A view of the scene with three planes from above. A light ray from a plane to the camera at position t intersects the image line at v. The camera is swiped from x_{01} to x_{02} with the shutter open (taken from [1])

A slice, $I_I(v)$, of the resultant swiped image is shown in Figure 3(a) and the corresponding EPI we aim to retrieve in Figure 3(b), plotted against the camera position, t. The slice, $I_I(v)$, is obtained by integrating the EPI over the range $x_{01} < t < x_{02}$:

$$I_I(v) = \int_{x_{01}}^{x_{02}} I(v, t)dt \tag{1}$$

where $I(v, t)$ is a horizontal slice of the EPI of the scene.

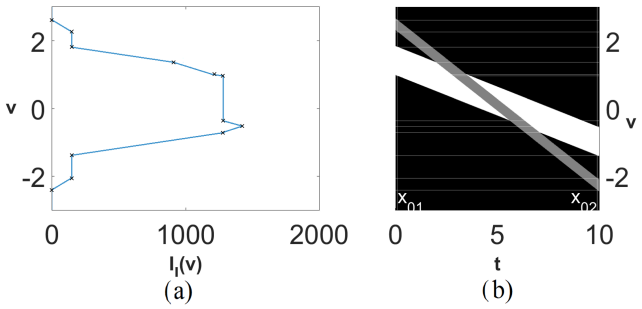


Fig. 3. A row of the swiped image, (a), created by integrating the EPI shown in (b) for a three plane image with constant plane texture surfaces. Each switchpoint is indicated by a white line in the EPI and by a cross in the row of the swiped image (taken from [1])

A "switchpoint" in the swiped image is created every time one plane occludes or disoccludes another in the EPI. This can be seen in Figure 3(b) where the darker plane occludes and disoccludes the lighter plane as the camera is swiped. The switchpoint locations in v are indicated in Figure 3(a) by crosses and in the EPI of Figure 3(b) by horizontal white lines. Switchpoints in the idealised case result in discontinuities in the horizontal gradient $\frac{\partial}{\partial v} I_I(u, v)$. In real images, switchpoints are detected by extrema in the second derivative of the horizontal gradient of $I_I(u, v)$. Each edge of a plane will result in a pair of switchpoints where it is occluded by the zeroth plane (i.e. at the left and right edges of Figure 3) whose separation is inversely proportional to the depth of the plane. By finding the extrema of the second derivative of $I_I(u, v)$, it is therefore possible to determine the depths of objects in the scene [1].

B. Algorithm Description

Leveraging the analysis of the previous section for the idealised plane case, we present an algorithm that is able to retrieve the EPI from a real swiped image. The algorithm described in this paper consists of the following steps illustrated in Figure 4. The depths of layers that are present within the swiped image are estimated (step (a) in Figure 4). This is achieved by finding peaks in the histogram of estimated disparities. The silhouette and location of the planes within the swiped image at each depth layer is then estimated (step (b) in Figure 4). An extension of the level set method which accounts for the amount of blur present within each depth layer is used to establish a silhouette of each plane at the start of the camera swipe. The surface texture of each plane at each depth layer is then recovered (step (c) in Figure 4) using the estimated depths, plane silhouettes and plane locations to deblur from the swiped image. Finally new images of the scene may be rendered from any camera location within the original swipe range (step (d) in Figure 4) using the estimated depths, plane surface textures and plane locations.

The general problem of recovering the plenoptic function from the swiped image arising from arbitrary camera motion is extremely challenging. In this paper we therefore make the

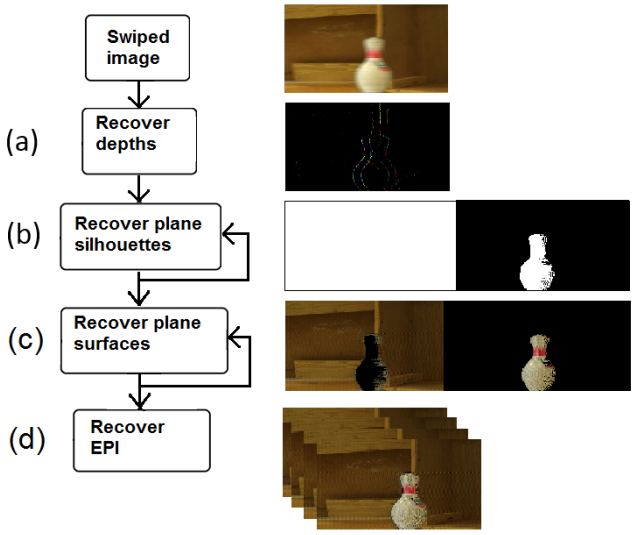


Fig. 4. A flowchart of the steps in the algorithm to recover an EPI from a swiped image. The switchpoints in the swiped image are shown in the image at step (a). The silhouettes of planes at each depth are shown at step (b). The surface of the planes at each depth are shown at plane (c).

problem more tractable by considering the case of uniform linear camera motion.

IV. EPI Recovery from a Swiped Image

A. Recovering depth layers

To recover the depths of layers present in the swiped image, we first analyse the switchpoints derived from the swiped image. As seen in Figure 3, a switchpoint is created in a given row in a swiped image every time a plane occludes or disoccludes another plane in the EPI as the camera is swiped. Switchpoints are found by detecting discontinuities in the horizontal intensity gradient of the swiped image. The occlusion of plane p by the zeroth plane will result in a pair of switchpoints separated by a distance, l_p, inversely proportional to the distance, z_p, of the plane from the camera. Switchpoints can be considered as forming an image of switchpoints, $M_s(u, v)$, which is used to infer scene geometry. The horizontal autocorrelation for each row of this image can be calculated:

$$R_{vv}(u, l) = \sum_{v \epsilon Z} M_s(u, v) M_s(u, v - l) \tag{2}$$

where M_s is the image of switchpoints, l is the lag and $R_{vv}(u, l)$ is the autocorrelation for each row of the image of switchpoints. We take the mean of the autocorrelation for each row to get the autocorrelation for the whole image, $\bar{R}_{vv}(l)$.

This is illustrated in Figure 5, which shows a swiped image in Figure 5(a) and the estimated image of switchpoints in Figure 5(b). The average autocorrelation, $\bar{R}_{vv}(l)$, of the rows of the switchpoint image is shown in Figure 5(c).

Peaks in the autocorrelation indicate the shift due to motion blur of the planes within the swiped image. The lag in (2) is also referred to as the disparity of the planes. Although planes occluding each other at the start or end of the camera swipe will supress the switchpoints of the more distant plane, we assume that sufficient switchpoints from each plane will remain to correctly estimate depths. The depths of the planes may be calculated from the disparity as:

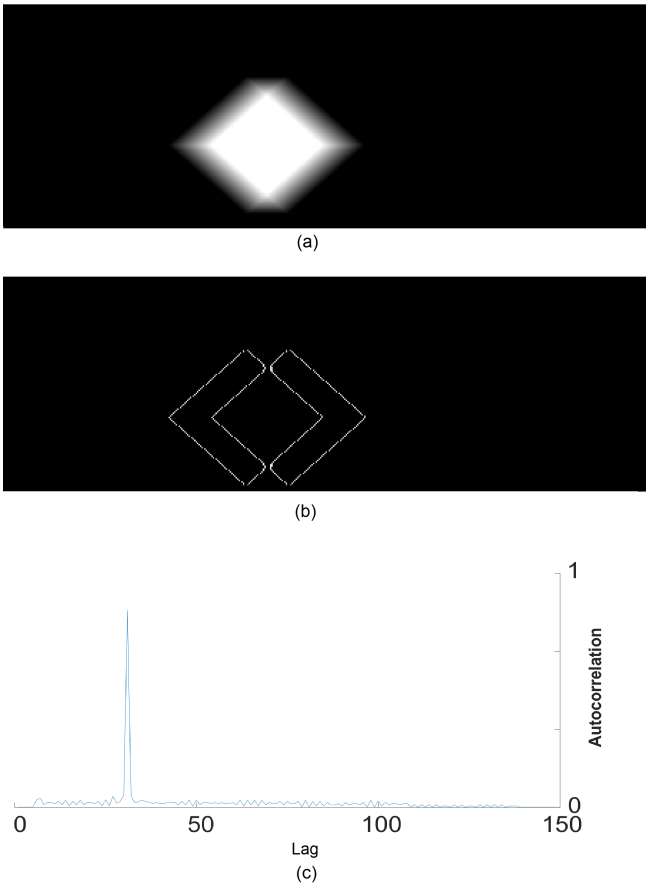


Fig. 5. The swiped image $I_I(u, v)$, shown in (a), leads to the image of switchpoints M_s, shown in (b), which contains the repeated outlines of planes. The autocorrelation histogram shown in (c) has a large peak for each depth.

$$z_p = f \frac{x_{01} - x_{02}}{l_p} \tag{3}$$

where l_p is the disparity of plane p at depth z_p within the swiped image, and f is the focal length of the camera. Using (3) requires knowledge of camera focal length and movement. From this, we then generate a histogram of estimated disparities within the image. The geometry of the planes within the scene are then calculated using trigonometry.

To increase the robustness of detecting the correct depth layers, a disparity likelihood distribution is created for each pixel using the method outlined in [23]. The range of the distribution is the calculated maximum and minimum disparities of the layers within the scene. This procedure is illustrated in Figure 6, where Figure 6(a) is the swiped image. The peak of the likelihood distribution for each pixel is plotted as an image in Figure 6(b). We sum the likelihood distributions for all the pixels in the swiped image to create an additional histogram of disparities within the image, shown in Figure 6(c). The histograms from Figure 6(c) and Figure 6(e) are normalised and added together to form a single histogram of disparities

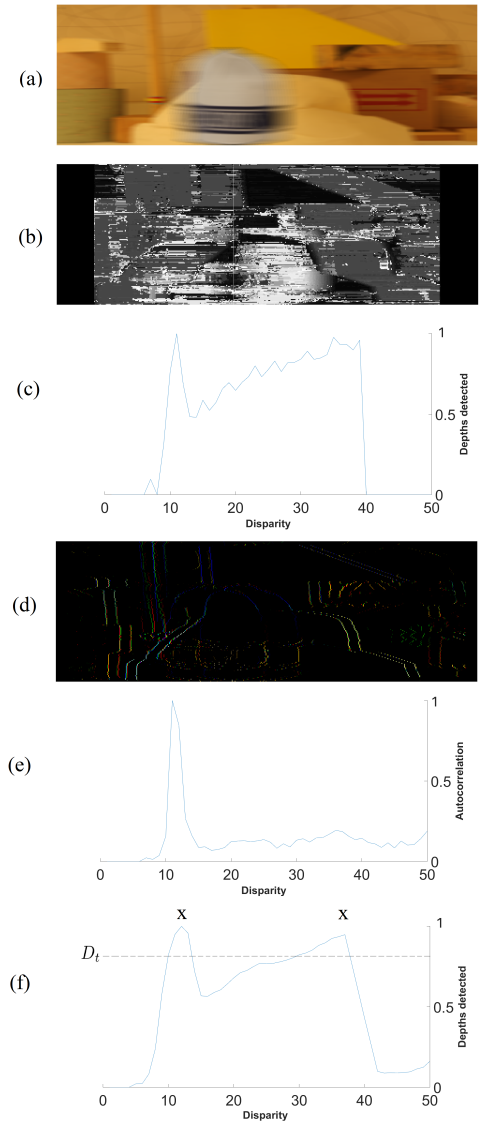


Fig. 6. The technique in [23] is applied to the simulated swiped image in (a). The peak of the depth distribution for each pixel forms an estimated depth map shown in (b). The histogram of estimated disparities is shown in (c), and is created by summing the likelihood distributions for all pixels within the swiped image. The image of switchpoints (d) is found from the swiped image, which contains repeated outlines of planes. The autocorrelation histogram shown in (e) has a peak for each disparity. The two histograms are combined and smoothed with a moving average filter in (f) to produce a peak at each disparity above the threshold D_t, indicated by a dotted line. The disparity of each peak, indicated by an x, determines the depth of the estimated layers in the swiped image.

detected within the image, shown in Figure 6(f), which is then smoothed by a moving average filter. Using this histogram, the number of depth layers is estimated as the number of peaks which are above a fraction, D_t, of the highest peak. The depths of these layers is determined by the location of these peaks. The choice of D_t is discussed in Section V.

To create a depth map of the swiped image that has only the detected layers, we retain only the likelihoods of the depth layers selected and classify each pixel according to the layer with the highest likelihood.

To improve the classification of the likelihood of each pixel being at a particular depth, we use a technique derived from Linear Discriminant analysis (LDA). This technique is used to build a model to find the common features of the pixels belonging to each depth layer. LDA is a well-known technique to extract features by dimension reduction, and has been used for applications such as image retrieval and face recognition [27], [28]. The features used in this technique are the values of red, green and blue colour channels (RGB) for the pixels within the colour swiped image, which is transformed into a probability of belonging to each of the depth layers.

A model is created representing the RGB colour intensity distribution within each layer. To identify the pixels that have a sufficiently high likelihood of being labelled correctly, only the pixels with a likelihood at one depth that is much greater than the likelihoods of belonging to other depths are chosen to build the model. Therefore, the inputs to build the model are a list of pixels, $\mathbf{x}_{p_i}$, labelled with the depth layer p to which they have been assigned. This is shown in Figure 7, where Figure 7(a) is the swiped image, and Figure 7(b) is the RGB value of pixels with labels represented by colour.

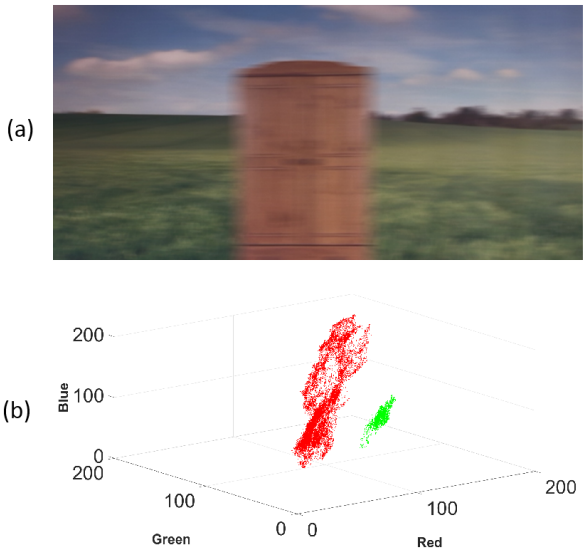


Fig. 7. Pixels of a swiped image containing two planes shown in (a). In (b) green coloured observations are pixels from the foreground plane and red coloured observations are pixels from the background plane. High likelihood labelled pixels are used to predict a model to classify other pixels within the image by depth.

We assume that the intensity distribution of pixels belonging to depth p follows a multivariate Gaussian distribution, which is characterised by the sample mean and covariance of the pixels for each label. The sample mean is calculated as [29]:

$$\boldsymbol{\mu}_p = \frac{1}{N_p} \sum_{i=1}^{N_p} I_I(\mathbf{x}_{p_i}) \tag{4}$$

where $\boldsymbol{\mu}_p$ is the mean of the pixels labelled with p, N_p is the number of pixels labelled p and $I_I(\mathbf{x}_{p_i})$ is the ith pixel labelled with p from the swiped image. The sample covariance for each label is calculated as [29]:

$$\mathbf{\Sigma}_p = \frac{1}{N_p - 1} \sum_{i=1}^{N_p} (I_I(\mathbf{x}_{p_i}) - \boldsymbol{\mu}_p)(I_I(\mathbf{x}_{p_i}) - \boldsymbol{\mu}_p)^T \quad (5)$$

where $\mathbf{\Sigma}_p$ is the covariance for label p.

A likelihood is assigned to each pixel in the swiped image that it belongs to each depth layer:

$$V(\mathbf{x}|p) = \frac{1}{(2\pi|\mathbf{\Sigma}_p|)^{\frac{1}{2}}} e^{-\frac{1}{2}(I_I(\mathbf{x})-\boldsymbol{\mu}_p)^T(\mathbf{\Sigma}_p)^{-1}(I_I(\mathbf{x})-\boldsymbol{\mu}_p)}. \quad (6)$$

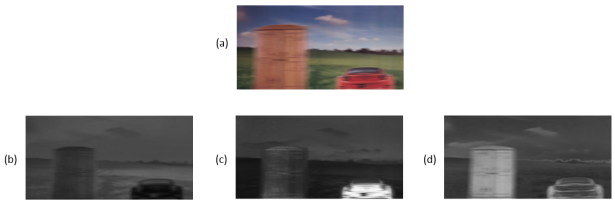


Fig. 8. Pixels of a swiped image containing three planes at different depths shown in (a). The probability pixels are at depth 3, the background, is shown in (b), the probability pixels are at depth 2 is shown in (c) and the probability pixels are at depth 1 is shown in (d).

The equation in (6) is used to transform each pixel in I_I, seen in Figure 8(a), into a likelihood of belonging to each depth layer, $V(\mathbf{x}|p)$, seen in Figure 8(b) - (d). Each pixel in each of the likelihood images is a scalar between 0 and 1, and an extended level set method operates on these images to estimate the silhouettes of planes at each depth. This transform makes the outline of each depth plane more reliable, as mislabelled pixels in the depth map are reclassified.

B. Recovering plane silhouettes

Once the depths of the layers present within the image have been found, the silhouettes of the planes at each depth are extracted. To obtain the EPI, the objects present within the scene that are at different distances from the camera must be segmented from each other. The use of the layer-based model makes depth ordering explicit, so we aim to find the silhouettes of each of the planes that represent objects at each depth within the scene. For convenience, we choose to recover the plane silhouettes when the camera is at position x_{02}, the rightmost end of the swipe.

For a sharp image, segmenting into regions of interest based on pixel intensity can be achieved using the level set method. The level set method was introduced by Osher and Sethian to track the movement of a front whose speed depends on the local curvature [30]. This front is defined as the locus of the zero level of a surface. The level set method has been used to segment images into different regions, in order to identify features of interest such as crystal growth [30], [31], flame propagation [30], [32] and cells [33].

Consider an image comprising of two regions which we wish to segment, as illustrated in Figure 9(a). To find the boundary, Γ, we define it as a function of ϕ, such that:

$$\begin{aligned} \phi(\mathbf{x}) < 0 &\leftrightarrow \mathbf{x} \in \Omega^- \\ \phi(\mathbf{x}) > 0 &\leftrightarrow \mathbf{x} \in \Omega^+ \\ \phi(\mathbf{x}) = 0 &\leftrightarrow \mathbf{x} \in \Gamma \end{aligned} \quad (7)$$

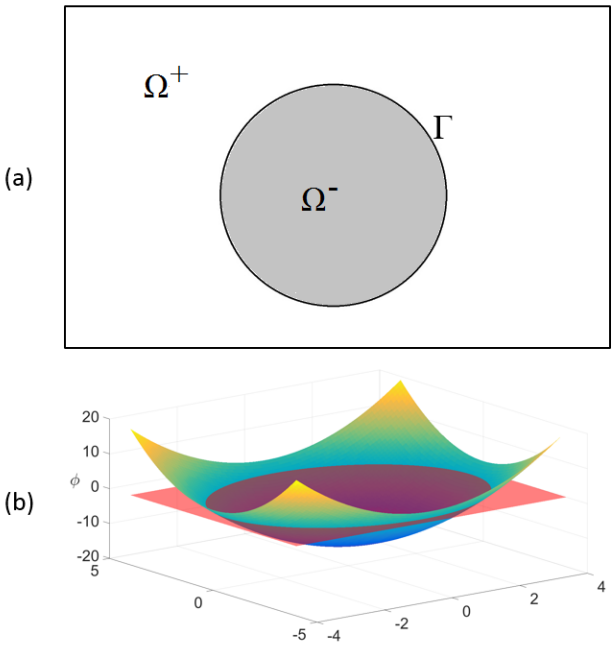


Fig. 9. An image segmented into two regions (a), and the surface ϕ that defines the boundary between the two regions (b) [31]

where $\mathbf{x}$ is a pixel index in the image. In this way, the two regions are defined by the single function, $\phi(\mathbf{x})$. The advantage of defining the regions Ω^- and Ω^+ using ϕ is that Γ is not explicitly defined, so regions can merge or split without hard decisions being made.

Typically the level set method searches for a boundary Γ between the regions, Ω^- and Ω^+, that minimises the energy of a function [34]:

$$E(\phi) = \int_{\Omega^+(\phi)} |V(\mathbf{x}) - c^+|^2 dx + \int_{\Omega^-(\phi)} |V(\mathbf{x}) - c^-|^2 dx \quad (8)$$

where $V(\mathbf{x})$ is a suitable function of the image that is to be segmented, $\mathbf{x}$ indexes the pixel, c^- is the mean of $V(x)$ within the $\Omega-$ region and c^+ is the mean of $V(x)$ within the $\Omega+$ region. Use of (7) demonstrates that a change in ϕ produces a change in the boundaries of Ω^+ and Ω^-, with (8) showing that this changes the energy of the segmentation.

To minimise the energy of (8), we evolve the function $\phi(\mathbf{x})$ over a time variable, τ, using a steepest descent method. The objective of the level set method is to find the segmentation of the two regions that results in the lowest possible energy.

It is shown in [34] that ϕ can be evolved by gradient descent to minimise $E(\phi)$ by choosing:

$$\frac{\partial \phi}{\partial \tau} = -\frac{\partial E}{\partial \phi}.$$

This PDE can be realized numerically by updating ϕ as follows:

$$\phi^{\tau+1} = \phi^{\tau} - \Delta\tau \frac{\partial E}{\partial \phi}$$

where [34]:

$$\frac{\partial E}{\partial \phi(\mathbf{x})} = -(|V(\mathbf{x}) - c^-(\phi^\tau(\mathbf{x}))|^2 - |V(\mathbf{x}) - c^+(\phi^\tau(\mathbf{x}))|^2)$$

where $\phi^\tau(\mathbf{x})$ is ϕ at iteration $\tau(\mathbf{x})$. At each iteration, τ, we calculate the updated areas of Ω^+ and Ω^- and then calculate the mean of $V(\mathbf{x})$ within each region to obtain c^+ and c^-.

When there is noise in the image, the level set algorithm may segment small, scattered regions which are not of interest. To counteract this, we penalise the mean curvature of $\phi(\mathbf{x})$, which is a local measure of the curvature of a surface for each pixel in ϕ. As a result, a region will not be segmented if the penalty caused by the increase in curvature of $\phi(\mathbf{x})$ outweighs the decrease in energy of (8). Following [31], [35], we define the mean curvature of $\phi(\mathbf{x})$ as:

$$\kappa = \nabla \frac{\nabla \phi}{|\nabla \phi|}.$$

Therefore, the update equation becomes [34]:

$$\phi^{\tau+1} = \phi^\tau - \lambda_1 \frac{\partial E}{\partial \phi} + \lambda_2 \kappa \tag{9}$$

where λ_1 and λ_2 are the weights attached to the level set energy and curvature respectively.

Given sufficient iterations, the image is segmented into two regions based on the intensity of the pixel values. To successfully use the level set function to obtain a region of interest, the image being segmented must have features in the regions that allow them to be distinguished from each other. Previous implementations of the level set function have used the intensity of the pixels [31].

An issue with the use of the canonical level set method for blurred images is that the blur introduced by camera motion with the shutter open produces a boundary between segments that is not sharp. Attempts to segment blurred images become more challenging as the blur within the images increases. We therefore modify the level set method to account for the blur at each depth to segment the swiped image into depth layers.

To segment the swiped image, we consider each of the depths independently. The plane that is the furthest away from the camera, z_P, is defined as the background plane, and is assumed to cover the entire background of the swiped image. Therefore, for this depth layer ϕ_P is uniformly negative. We assume for now that the planes z_1 to z_{P-1} do not occlude each other, but occlude only the background. The case of planes occluding each other is discussed in Section IV-D.

For each non-background plane, we model the situation as the plane at that depth, z_p, swiping over only the background plane. All other planes are considered as part of the background. We define a different level set surface $\phi_p(\mathbf{x})$ to determine the silhouette of each non-background plane. To achieve that we introduce the following change to the calculation of energy for the level set method, defining the energy of the image at depth p as:

$$E_p(\phi_p(\mathbf{x})) = \lambda_2 \kappa_p + \lambda_1 \int |V(\mathbf{x}|p) - V_{c_p}(\phi_p(\mathbf{x}))|^2 dx \tag{10}$$

where $V_{c_p}(\mathbf{x})$ is the best approximation for the image $V(\mathbf{x}|p)$ using two regions with set values, blurred with a known blurring kernel that depends on the depth of the plane.

The approximation of $V(\mathbf{x}|p)$ is obtained from $\phi_p(\mathbf{x})$ as follows: we first introduce

$$V_{\phi_p}(\mathbf{x}) = \begin{cases} 1 & \phi_p(\mathbf{x}) < 1 \\ 0 & \text{otherwise} \end{cases}$$

where $V_{\phi_p}(\mathbf{x})$ is the current estimate of the silhouette of the depth layer given ϕ_p. Moreover,

$$V_{b_{p-}}(\mathbf{x}) = h_p * V_{\phi_p}(\mathbf{x})$$
$$V_{b_{p+}}(\mathbf{x}) = h_p * (1 - V_{\phi_p}(\mathbf{x}))$$

where h_p is a blurring kernel that is a box function with a length determined by that of the depth of the layer being segmented.

The value of $\mathbf{c}_{b_p}$ is chosen to minimise:

$$\|\mathbf{V}_{b_p}\mathbf{c}_{b_p} - \mathbf{v}_p\|^2$$

where

$$\mathbf{V}_{b_p} = \begin{bmatrix} V_{b_{p-},1} & V_{b_{p+},1} \\ V_{b_{p-},2} & V_{b_{p+},2} \\ \vdots & \vdots \\ V_{b_{p-},n} & V_{b_{p+},n} \end{bmatrix}$$

and

$$\mathbf{c}_{b_p} = \begin{bmatrix} c_{b_{p-}} \\ c_{b_{p+}} \end{bmatrix}$$

where n is the number of pixels, $V_{b_{p-},1}, V_{b_{p-},2}$ ... $V_{b_{p-},n}$ are the elements of $V_{b_{p-}}(\mathbf{x})$ and $V_{b_{p+},1}, V_{b_{p+},2}$ ... $V_{b_{p+},n}$ are the elements of $V_{b_{p+}}(\mathbf{x})$, and $\mathbf{v}_p$ is the elements in $V(\mathbf{x}|p)$ formed into a column vector. Finally, the best approximation to $V(\mathbf{x}|p)$ is given by:

$$\mathbf{V}_{c_p} = \mathbf{V}_{b_p}\mathbf{c}_{b_p}$$

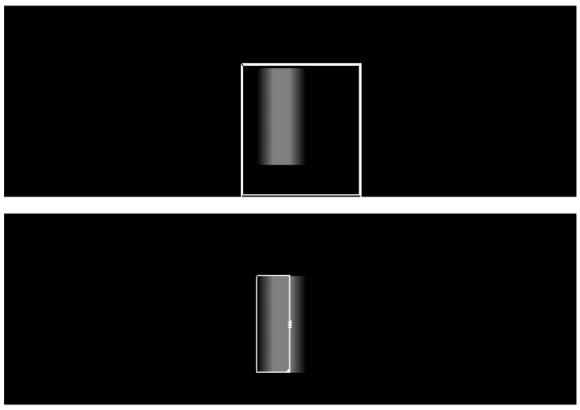


Fig. 10. (a) shows the initial ϕ boundary for a blurred object and (b) shows this object being successfully segmented. The white outline segments the plane silhouette for the rightmost camera position.

Given the equation for the energy of the segmentation, (10), the change in energy due to a change in the value of $\phi_p(\mathbf{x})$ is determined. For each pixel, the value of $\phi_p(\mathbf{x})$ is changed to $-\phi_p(\mathbf{x})$, thereby reversing the sign of $\phi_p(\mathbf{x})$ for

that pixel and changing the boundary. Given the new boundary, the new energy is calculated. This gives a numerical method for estimating the partial derivative:

$$\frac{\partial E_p}{\partial \phi_p}(\mathbf{x}) = \frac{E_p(\phi_{+_p}(\mathbf{x})) - E_p(\phi_{-_p}(\mathbf{x}))}{|\phi_p(\mathbf{x})|} \tag{11}$$

where $\phi_{+_p}(\mathbf{x})$ is the ϕ_p at the current iteration where the value of ϕ_p at $\mathbf{x}$ is positive and $\phi_{-_p}(\mathbf{x})$ is the ϕ_p at the current iteration where the value of ϕ_p at $\mathbf{x}$ is negative.

In principle the energy of the entire image must be calculated for every pixel being considered. To reduce the amount of calculation required, it is assumed that only pixels within a predefined distance of the boundary are likely to reduce the energy with a sign change, and so only these pixels need to be considered. This is known as the narrow band level set [36].

Using (11) in (9), we evolve each non-background ϕ_p, as in Figure 10, for a set number of iterations to estimate the plane silhouette for each non-background plane.

C. Recovering plane surfaces

Once the depths in the scene have been found using the method in Section IV-A, and the plane silhouettes and locations using the method in Section IV-B, the texture surfaces of the planes may be recovered.

For the simplest scene type, in which there is only a single plane, a sharp image of the plane can be turned into the swiped image by a simple linear convolution of the sharp plane image with a 1D box function with a length equalling the disparity of the plane due to the depth. Therefore, to recover the surface of the original plane from the swiped image, the well-studied techniques in the field of deblurring will, for this case, produce a good result.

When we consider a more realistic scene comprising multiple planes, the process of recovering the plane surfaces becomes more complicated. This is because as the camera is swiped, planes at different depths will have different disparities in the swiped image and may occlude each other due to their positioning within the scene. The effect of these occlusions turns the problem of recovering plane surfaces from a linear one to a non-linear one, making the recovery more difficult. To recover the plane surfaces, we first consider a situation including planes of different depths which do not occlude each other.

Planes within the scene at different depths will have a disparity during the camera swipe inversely proportional to their depth, and will have different occlusions depending on depth, so are recovered separately. To get the forward model of the effect of camera swiping, we define a matrix $\mathbf{A}_p$ that when multiplied by a row of the sharp image of a plane will produce a row of the swiped image of the plane:

$$\mathbf{A}_p \tilde{\mathbf{v}}_p = \mathbf{I}_{I,p} \tag{12}$$

where $\mathbf{A}_p$ is the swiping matrix, measuring $n \times m$, where n is the width of the swiped image, m is $n + b$, where b is a buffer the size of the maximum disparity expected in the swiped image, to account for planes wider than the width of the image. $\tilde{\mathbf{v}}_p$ is a transposed row of the sharp image of plane p measuring m by 1 and $\mathbf{I}_{I,p}$ is a transposed row of the swiped image of plane p measuring n by 1.

The $\mathbf{A}_p$ matrix in (12) is populated with the following elements to model the effect of swiping the camera:

$$a(i,i) = \begin{cases} 0.5(2L - (L-M)^2)/L & \text{if } M = 0 \\ 0.5/L & \text{if } M > 0 \end{cases}$$

$$a(i,i-1) = \begin{cases} 0.5(L-M)^2/L & \text{if } M = 0 \\ 0 & \text{if } M > 0 \end{cases}$$

$$a(i,j) = \begin{cases} 0 & \text{if } M = 0 \\ 1/L \text{ for } i - M < j < i & \text{if } M > 0 \end{cases}$$

$$a(i,i-M) = \begin{cases} 0 & \text{if } M = 0 \\ \frac{0.5(1+2L-2M-(L-M)^2)}{L} & \text{if } M > 0 \end{cases}$$

$$a(i,i-M-1) = \begin{cases} 0 & \text{if } M = 0 \\ \frac{0.5(L-M)^2}{L} & \text{if } M > 0 \end{cases}$$

where L is the disparity, M is floor(L), i is the row index, and j is the column index.

To find the forward model of a multiple-plane scene, we concatenate the $\mathbf{A}_p$ matrices and $\tilde{\mathbf{v}}_p$ vectors for each plane to form:

$$\mathbf{A} = [\mathbf{A}_1, \mathbf{A}_2 ... \mathbf{A}_P] \tag{13}$$

and

$$\tilde{\mathbf{v}} = [\tilde{\mathbf{v}}_\mathbf{1}^T; \tilde{\mathbf{v}}_\mathbf{2}^T; ... \tilde{\mathbf{v}}_\mathbf{P}^T]^T \tag{14}$$

where $\mathbf{A}$ is the swiping matrix for the entire scene and $\tilde{\mathbf{v}}$ is a concatenated row of the sharp image of planes at each depth layer. Use of these matrices gives the equation:

$$\mathbf{A}\tilde{\mathbf{v}} = \hat{\mathbf{I}}_I \tag{15}$$

where $\hat{\mathbf{I}}_I$ is defined as:

$$\hat{\mathbf{I}}_\mathbf{I} = \sum_{p=1}^{P} \mathbf{I}_{I,p} \tag{16}$$

and represents the swiped image from the planes in the absence of the effects of occlusions.

The difficulty of recovering the scene is due to the non-linear effects of the occlusions on the swiped image. To account for the occlusions of the planes on the swiped image, we modify the $\mathbf{A}_p$ matrices in depth order, taking advantage of the explicit depth ordering of the layer-based model. $\mathbf{A}_1$ is related to the closest plane to the camera, and so cannot be occluded by another plane, and remains unchanged. The entries of $\mathbf{A}_2$ which are calculated to be occluded by plane 1, by our knowledge of plane silhouette and location from Section IV-B, are deleted. This process is repeated for each layer in turn, with the elements occluded by planes closer to the camera changed to zero. The $\mathbf{A}$ matrix for the situation where the effect of occlusions is disregarded is shown in Figure 11(a). The $\mathbf{A}$ matrix incorporating the effects of occlusions is shown in Figure 11(b), where some of the elements for the plane further from the camera have been changed to zero due to occlusion.

Fig. 11. (a) shows the $\mathbf{A}$ matrix for a scene with two planes without considering the effects of occlusion. (b) shows $\mathbf{A}$ for the same scene when occlusion is considered. The elements for the closer plane (on the right), are unchanged, but some elements for the further plane (on the left) are set to zero.

The forward model of using (15) to obtain a row of the swiped image is demonstrated in Figure 12. A row of the image in Figure 12(a) is generated from the multiplication of the $\mathbf{A}$ matrix with the concatenated sharp images of planes, $\tilde{\mathbf{v}}$, shown in Figure 12(b).

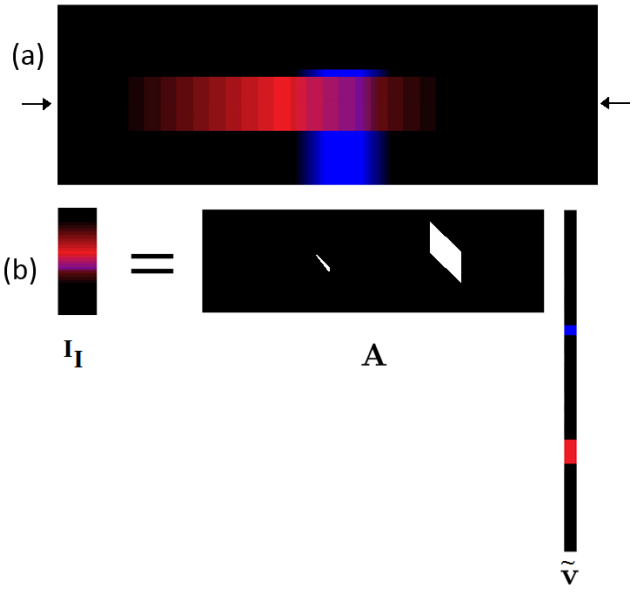


Fig. 12. Each row of the swiped image, I_I shown in (a), can be formed by a row of the sharp image of each plane in $\tilde{\mathbf{v}}$ multiplied by $\mathbf{A}$. The resultant row of the swiped image created in (b) is indicated in (a) by arrows. In this figure, there are two depths present within the image, related to a red and blue plane shown within $\tilde{\mathbf{v}}$.

The $\mathbf{A}$ matrix is calculated row by row due to variations within the scene geometry with height. Using the minimum solution to the following equation, we estimate each row of the original images of the planes from the swiped image using:

$$\|\mathbf{A}\tilde{\mathbf{v}} - \mathbf{I}_I\|^2 = 0. \tag{17}$$

The equation in (17) is not well posed due to the low pass filtering effects of swiping the camera. In situations such as this, standard least squares estimation can produce unwanted effects due to the large norm of the least squares solution. To obtain the best estimate of the original planes, we use the regularisation method of the iterative shrinkage thresholding algorithm (ISTA) to stabilize the solution [37], [38].

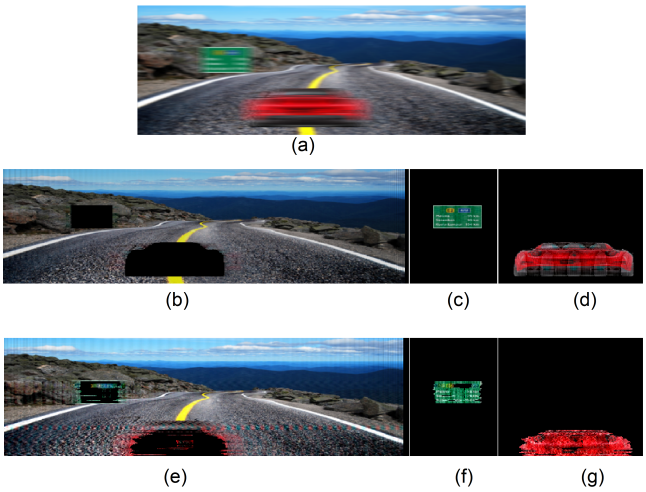


Fig. 13. Image of recovered planes at depth 3 (b), depth 2 (c), and depth 1 (d), from swiped image (a) when the depths and planes outlines are correct. When plane silhouettes are incorrect the recovered planes at depth 3 (e), depth 2 (f), and depth 1 (g) are incorrect.

D. Plane recovery optimisation

The technique outlined above can be used to recover the plane surfaces within the image, given an estimate of the plane silhouettes from Section IV-B. In this section, we propose a method of further refining the recovery of the planes. The reasons for this are twofold, with the first being that the assumption from Section IV-B, that non-background planes only occlude the background and not each other, is not generally true in real world images. Therefore, we get a good result when non-background planes do not overlap, but inaccuracies in the plane silhouettes when they do. The second is that the more accurate the plane silhouettes are, the more accurate the recovered plane surfaces and therefore recovered EPI will be. When the planes in the image are segmented wrongly, the content of the planes will leak into each other in $\tilde{\mathbf{v}}$, as can be seen in Figure 13(e). This leakage does not occur when plane silhouettes are correct, such as in Figure 13(b). Minimising the error in the boundary estimate will minimise this leakage.

The level set method in Section IV-B operates on a transform of the swiped image, the probability of each pixel belonging to each depth. When non-background planes occlude each other in the swiped image, then a pixel can belong to more than one plane. Use of the algorithm from Section IV-B on the 3-plane swiped image in Figure 14(a), will result in incorrect plane silhouettes where the planes overlap, resulting in the recreated swiped image shown in Figure 14(b).

To solve the problem of overlapping planes, we use the level set method again, but now calculate ϕ using the recovered planes. This method uses the recovered planes, where a change

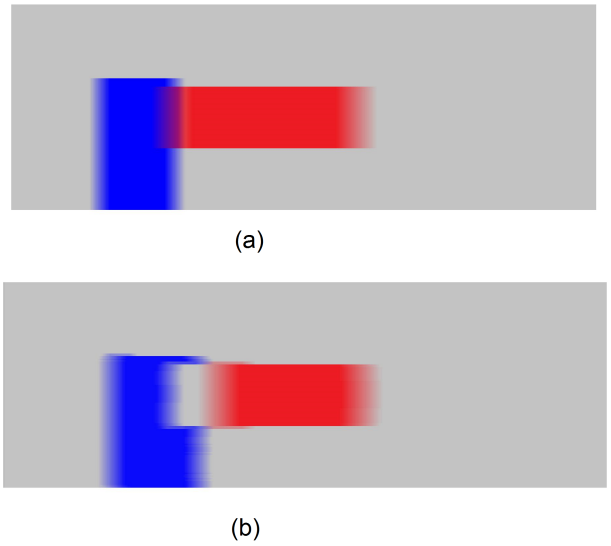


Fig. 14. The recreated swiped image from the recovered planes (b) from the swiped image (a) are estimated wrongly due to occlusion due to the swipe.

in the silhouette of a plane results in a different plane surfaces at other depths due to occlusion. Therefore, the ϕ for each depth cannot be considered separately, and so ϕ is defined as:

$$\phi = [\phi_1, \phi_2, ...\phi_P]$$

where ϕ_1, ϕ_2 and ϕ_P are the level set surfaces for depth layers 1,2 and P respectively. As the recovered planes are separated by depth, a pixel in the swiped image can belong to any combination of planes. For this method, we use the RGB value of a pixel $\tilde{\mathbf{v}}(\mathbf{x})$ in a recovered plane, and compare it to the RGB distributions of the planes at each depth. If the recovered pixel closely matches the colours of the planes at its depth, then we assume the pixel is correctly part of the plane silhouette. If the recovered pixel matches more closely the colours in a plane at another depth, we assume it is not part of the plane silhouette.

To assess how closely the colour of the pixel matches the colour of the plane around it, we calculate the mean and sample covariance of the pixels in a window around the pixel:

$$\boldsymbol{\mu}_L(\mathbf{x}) = \frac{1}{N_L}\sum_{i=1}^{N_L}\tilde{\mathbf{v}}(\mathbf{x}_{L,i}) \tag{18}$$

where $\boldsymbol{\mu}_L(\mathbf{x})$ is the mean of the pixels, where $\phi(\mathbf{x})$ is negative, in a window around $\mathbf{x}$, N_L is the number of pixels, where $\phi(\mathbf{x})$ is negative, in a window around $\mathbf{x}$ and $\tilde{v}(\mathbf{x}_{L,i})$ is the ith pixel, where $\phi(\mathbf{x})$ is negative, in a window around the pixel $\mathbf{x}$. The sample covariance for each label is calculated by [29]:

$$\boldsymbol{\Sigma}_L(\mathbf{x}) = \frac{1}{N_L - 1}\sum_{i=1}^{N_L}(\tilde{\mathbf{v}}(\mathbf{x}_{L,i}) - \boldsymbol{\mu}_L(\mathbf{x}))(\tilde{\mathbf{v}}(\mathbf{x}_{L,i}) - \boldsymbol{\mu}_L(\mathbf{x}))^T \tag{19}$$

where $\boldsymbol{\Sigma}_L(\mathbf{x})$ is the covariance of the pixels, where $\phi(\mathbf{x})$ is negative, in the window around $\mathbf{x}$. We then calculate the likelihood the pixel at $\mathbf{x}$ matches the distribution:

$$M_L(\mathbf{x}) = \frac{1}{(2\pi|\boldsymbol{\Sigma}_L(\mathbf{x})|)^{\frac{1}{2}}} e^{-\frac{1}{2}(\tilde{\mathbf{v}}(\mathbf{x})-\boldsymbol{\mu}_L(\mathbf{x}))^T(\boldsymbol{\Sigma}_L(\mathbf{x}))^{-1}(\tilde{\mathbf{v}}(\mathbf{x})-\boldsymbol{\mu}_L(\mathbf{x}))} \tag{20}$$

where $M_L(\mathbf{x})$ is the likelihood the pixel at $\mathbf{x}$ matches distribution defined by $\boldsymbol{\Sigma}_L(\mathbf{x})$ and $\boldsymbol{\mu}_L(\mathbf{x})$. The energy at the pixel is defined as:

$$E_L(\mathbf{x}) = -\ln(M_L(\mathbf{x})). \tag{21}$$

To find the likelihood the pixel matches the planes at other depths, we consider the recovered planes as being composed of the recovered plane at each depth, as in (14). We repeat (18), (19), (20) and (21) for a window with the same coordinates as $\mathbf{x}_{L,i}$ for each depth $\tilde{\mathbf{v}}_1$, $\tilde{\mathbf{v}}_2$ ... $\tilde{\mathbf{v}}_P$ to obtain $E_{L,1}$, $E_{L,2}$ ... $E_{L,P}$, a measure of the energy of the pixel at $\mathbf{x}$ against the pixels in a window at each depth.

To evolve the level set function we calculate the gradient:

$$\frac{\partial E(\mathbf{x})}{\partial \phi(\mathbf{x})} = -\frac{(E_{L,p=d} - \min(E_{L,p\neq d}))}{|\phi(\mathbf{x})|} \tag{22}$$

where d is the depth that pixel $\mathbf{x}$ belongs to in $\phi(\mathbf{x})$, $E_{L,p=d}$ is the energy from the depth that $\mathbf{x}$ belongs to and $\min(E_{L,p\neq d})$ is the minimum energy from a depth that $\mathbf{x}$ does not belong. To calculate (22), for each pixel $\phi(\mathbf{x})$ where the sign is positive we must recalculate the $\mathbf{A}$ matrix and therefore the recovered planes for when the sign of $\phi(\mathbf{x})$ is negative to compare the colours of the recovered pixel.

As in Section IV-B, ϕ is evolved iteratively using a steepest descent algorithm to find plane silhouettes and therefore plane surfaces. We initialise ϕ using the plane silhouettes in Section IV-B. After each iteration $\mathbf{A}$ is recalculated based on ϕ and the recovered planes are re-estimated.

Creating the LDA for each plane relies on having a good initial estimate of plane boundaries, so the plane silhouettes must be close to correct answer before this procedure is applied. Furthermore, a different $\tilde{\mathbf{v}}$ needs to be recovered for each pixel within each depth, creating a high amount of complexity. The fast marching method is used to limit the pixels tested to near the boundaries, limiting the speed of the boundary change.

E. EPI recovery and image rendering

Once the surface intensities of the planes have been obtained, this information is combined with the knowledge of the plane silhouettes and plane geometry from the switchpoints to reconstruct the EPI of the scene. To render an image for a novel camera position, we select the new camera distance from x_{02}, and use (3) to find the shift of the location of the planes within the new image. Once the planes have been shifted, we create the new image by copying the pixels where $\phi(\mathbf{x})$ is negative from each depth layer into the corresponding pixels of the new image. This process repeats for each depth, beginning with the furthest plane, to account for occlusions. The full EPI recovery algorithm is summarised in Algorithm 1.

Algorithm 1 Compute EPI from swiped image

1: Find switchpoints in swiped image (Section IV-A)
2: Calculate horizontal autocorrelation of image to estimate number of layers and layer depths using (2)
3: Transform pixels in swiped image to probabilities for each depth using (6)
4: Initialise plane silhouettes
5: **for** Iterations less than threshold **do**
6: Calculate (11) using depth probabilities and evolve plane silhouettes using extension to level set method
7: **end for**
8: **for** Iterations less than threshold **do**
9: Evolve plane silhouettes using recovered plane surfaces using (22)
10: **end for**
11: Recreate EPI using scene geometry from depths, plane positions and plane surfaces

V. Results

To evaluate the algorithm proposed in Section IV, we tested it on both simulated and real images. To create real swiped images, fronto-parallel images at different depth layers were arranged to form a diorama that resembles a real life scene. A CineMoco camera slider was used to create a swiped image by moving a DSLR camera between two points with the shutter open.

The number of depth layers was calculated as in Section IV-A. The parameter D_t was chosen empirically as $D_t = 0.7$. If the value is too high then too few or zero depths are detected. If the parameter is set too low then spurious depths are detected. This is illustrated by Figure 15, where a swiped image in Figure 15(a) produces the depth probability histogram in Figure 15(b). The number of depths detected from the image varies with the value of D_t, as shown in Figure 15(c). For a scene with no fronto-parallel planes, there is no correct answer for the number of layers, so the number of layers detected will be quantised depending on the value of D_t. The boundary of the planes in the swiped image was segmented using a ϕ defined by the method proposed in Section IV-B, as illustrated in Figure 16. The level set method is initialised with multiple segments, as shown in Figure 16(a), in which the initial boundary contains a set percentage of the energy within each depth. The boundary is then evolved using the level set method, resulting in the plane boundary shown in Figure 16(b).

The run time of the algorithm on MATLAB running on a computer with an Intel i7-3770 CPU and 8GB of RAM when two depth layers were present was approximately 6 minutes for a swiped image of dimension 450 by150 pixels. Most of the complexity of the algorithm comes from the optimisation step described in Section IV-D, as the plane surfaces have to be estimated for each change in the sign of $\phi(\mathbf{x})$. As the optimisation step is run on each depth layer, the complexity of the algorithm is proportional to $O(P)$.

The results of the algorithm on a dataset of simulated images are shown in Figure 17, and with real images in Figure 18,

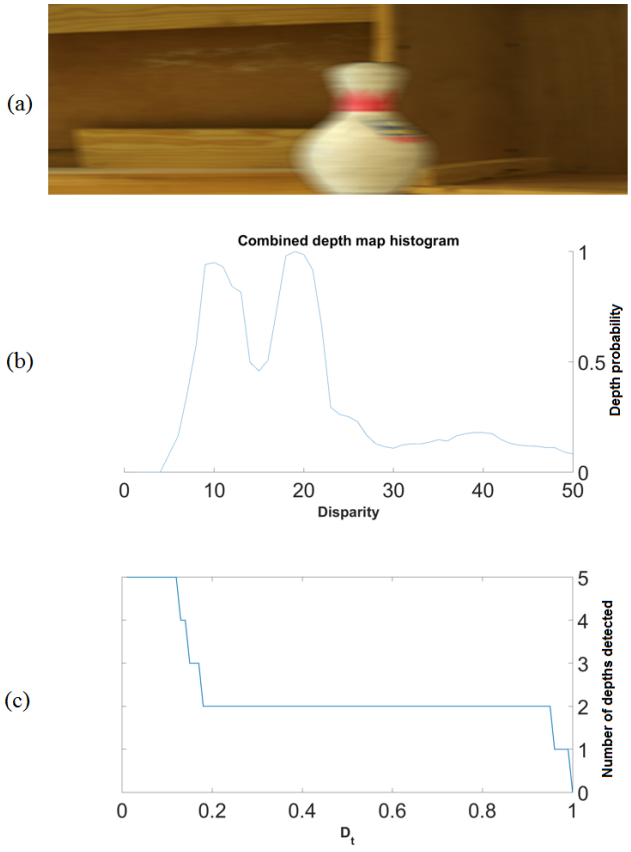


Fig. 15. (a) shows a swiped image with two layers. (b) shows the depth probability at each disparity using the technique in Section IV-A. (c) shows how the number of depth layers detected varies with D_t for this image. The value of D_t is set empirically, as if it is too high then only a single layer is detected, and if it is too low then spurious depths are detected.

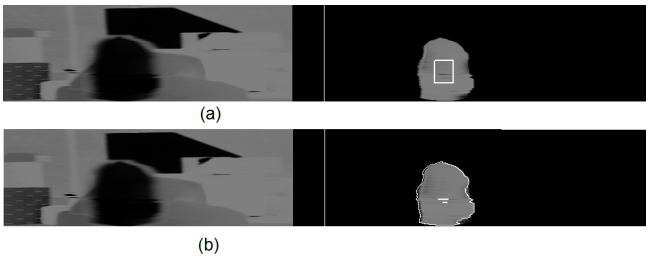


Fig. 16. (a) shows the initial guess of Γ as a white rectangle, and (b) shows the Γ for level set method extension after 20 iterations. The leftmost plane is a background plane, so ϕ is always negative in that area.

demonstrating the validity of the algorithm under a variety of scenes.

The depth maps that we estimate using the proposed technique are shown in Figure 19. Figure 19(a) shows the swiped images, with Figure 19(b) showing the ground truth depth map from the mid-point of the camera swipe. Figures 19(c) and (d) compare the results from [23] with the depth map resulting from the mid-point in the camera swipe of the recovered EPI. As the technique in this paper uses the algorithm from [23] to obtain an initial estimate of the depth in the image, the improved depth map from our technique justifies the iterative updates of the depth silhouettes using colour information.

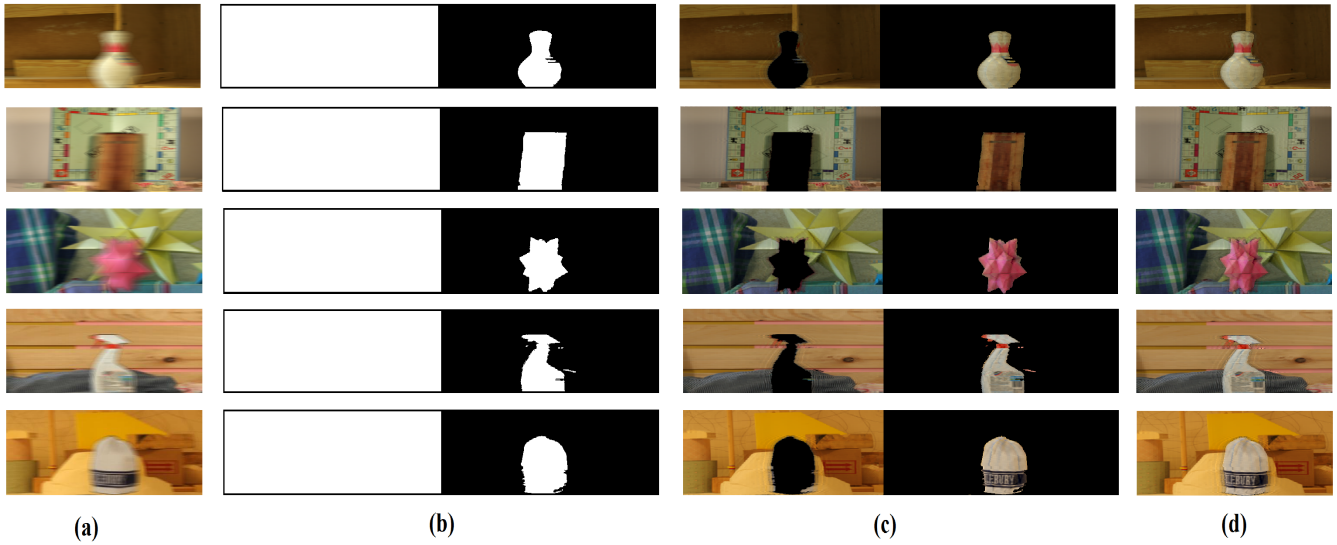


Fig. 17. Column (a) are simulated swiped images, which are made from two planes at different depths, constructed from cropped planes in [39], [40]. Column (b) is the estimated plane silhouettes, in descending depth order, with column (c) being the recovered plane surfaces. From the recovered planes, we recover the EPIs of the scenes, and so can create novel views of the scene for different camera position, such as in column (d), which shows the scene captured at the mid-point of the swipe.

Fig. 18. Column (a) are real swiped images, which are made from two planes at different depths to form a diorama, constructed from cropped planes in [39], [40]. Column (b) is the estimated plane silhouettes, in descending depth order, with column (c) being the recovered plane surfaces. From the recovered planes, we recover the EPIs of the scenes, and so can create novel views of the scene for different camera position, such as in column (d), which shows the scene captured at the mid-point of the swipe.

Given a swiped image (e.g. Figure 20(a)), the depths in the image can be detected using the switchpoints (Figure 20(b)). The plane surfaces can then be recovered (Figure 20(c)), allowing the EPI to be recovered, and new images of the scene generated for arbitrary intermediate camera positions (Figure 20(d) and (e)).

In the recovered planes, areas which are always occluded during the swipe cannot be recovered as they are never seen. This can be seen in Figure 20(c), where those parts of the background plane that are never visible in Figure 20(a) cannot be recovered, leading to a hole in the recovered background plane. These holes are smaller in width than the objects occluding them due to the motion of planes during the swipe. Slices of the recovered EPI from the two upper examples of Figure 17 are shown in Figure 21, demonstrating that the algorithm presented in this paper successfully fulfils the goal of recovering the EPI of a scene from a single swiped image.

A further set of recovered EPIs can be seen in Figure 22 where there are three planes present within the scene. The scenes shown in Figure 22 have 3 depths present within the image. Some of the pixels in the final example in Figure 22 include contributions from all three planes. This can be seen from the leftmost panel of image Figure 22(c) which includes recovered background pixels that lie between the two foreground objects. This would not be possible using the traditional methods of segmenting and deblurring by depth. The results of the algorithm on a simulated swiped image with four depths is shown in Figure 23. Simulated images are used in this case, as it is challenging to get all the planes within focus for a real image with four depth layers, whilst having enough of a difference in disparity to enable detection of each layer.

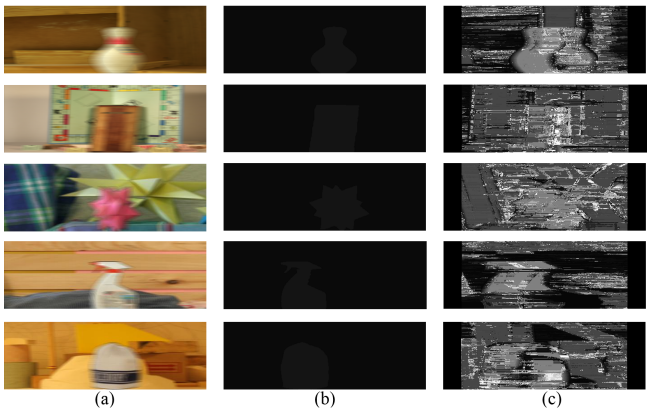


Fig. 19. (a) are simulated swiped images, which are made from two planes at different depths to form a diorama, constructed from cropped planes in [39], [40]. (b) are the ground truth depth maps, from the mid-point of the camera swipe. (c) are the estimated depth maps using the technique from [23], and (d) are the estimated depth maps using the technique in this paper at the mid-point of the camera swipe as a comparison of techniques.

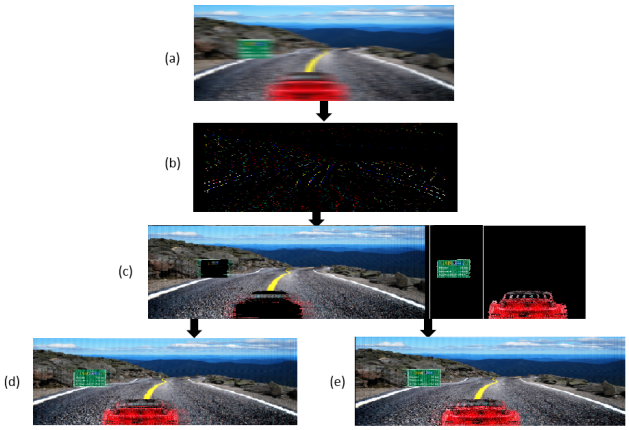


Fig. 20. Creation of new views of the scene from a generated swiped image. (a) shows a swiped image of a scene, (b) shows an image of the detected switchpoints, (c) shows the recovered planes from furthest on the left to closest on the right, with each depth separated by a white line, and (d) and (e) show new views of the scene rendered from different perspectives.

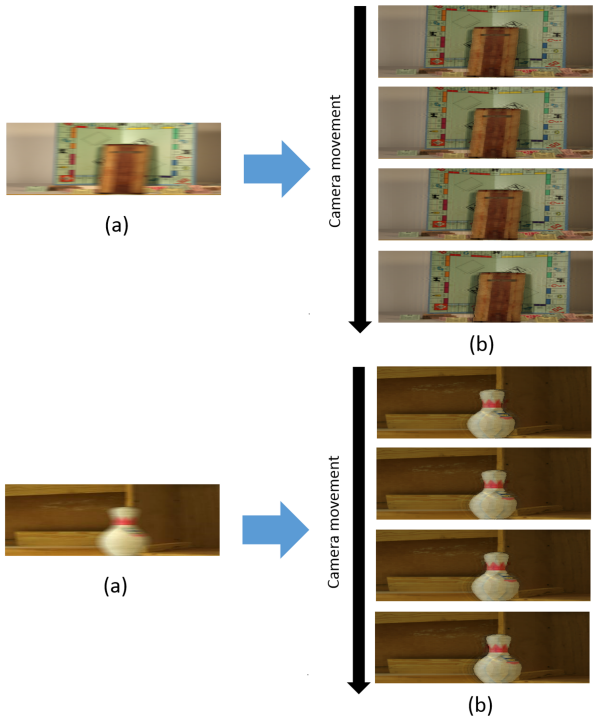


Fig. 21. The EPI slices denoted by (b) are recovered from the swiped images denoted by (a) using the proposed algorithm from Section IV.

TABLE I
COMPARISON OF RECOVERED IMAGES OF THE SCENE AGAINST GROUND TRUTH IMAGES.

Swiped image	Layers disparity	Recovered EPI SSIM	Recovered swiped image SSIM	Recovered EPI MSE	Recovered swiped image MSE
Pin	5 10	0.993	1.000	14.25	0.0443
Pin	10 20	0.988	1.000	27.64	0.0587
Pin	15 30	0.985	1.000	34.84	0.0703
Monopoly	5 10	0.982	1.000	22.99	0.0885
Monopoly	10 20	0.969	1.000	40.33	0.1453
Monopoly	15 30	0.958	1.000	58.51	0.1825
Moebius	5 10	0.975	1.000	34.55	0.0920
Moebius	10 20	0.957	1.000	64.69	0.1411
Moebius	15 30	0.939	1.000	89.53	0.1938
Laundry	5 10	0.964	0.998	50.30	0.1899
Laundry	10 20	0.904	0.993	103.07	0.2398
Laundry	15 30	0.882	0.995	134.48	0.2774
Lampshade	5 10	0.982	1.000	64.52	0.2264
Lampshade	10 20	0.970	0.999	110.39	0.2274
Lampshade	15 30	0.959	1.000	158.33	0.2571

This demonstrates that the technique can be extended to more complex scenes that include multiple objects, as would be typically seen in real world swiped photographs.

The structural similarity (SSIM) index [41], using the default settings of the MATLAB implementation, was used to compare the recovered images in the EPI, and the recovered swiped image from the created EPI, to the ground truth synthetic images. The results of the proposed EPI recovery algorithm are shown in Table I. The third column gives the SSIM of the recovered EPI while the fourth column gives the SSIM of a reconstructed swiped image. As a comparison, the Mean Squared Error (MSE) of the recovered EPIs and reconstructed swiped imaged are the fifth and sixth columns. To create the reconstructed swiped image, the recovered EPI was integrated between x_{01} and x_{02}. It can be seen that, although there are small errors in the recovered EPI, the recreated swiped image is, in all cases, almost perfect.

Table I shows results for different swipe lengths corresponding to maximum disparities of 10, 20 or 30 pixels. The results of this are plotted in Figure 24. From the simulations performed, it can be seen that there is a trade-off between having a larger camera swipe, which enables an EPI which encompasses more camera locations, and having an EPI with more accuracy. This makes intuitive sense, since with all deblurring algorithms, the larger the amount of blur the harder it is to recover a sharp image.

Figure 25 compares the proposed method with a recent blind deblurring algorithm [43]. Although [43] creates a view of the scene from only a single perspective, as shown in Figure 25(d), it can be compared with a rendered image from the recovered EPI using the technique from this paper, as shown in Figure 25(b) as well as with the ground truth in Figure 25(c). An attempt to create an EPI from a single deblurred image would have the disadvantage against the

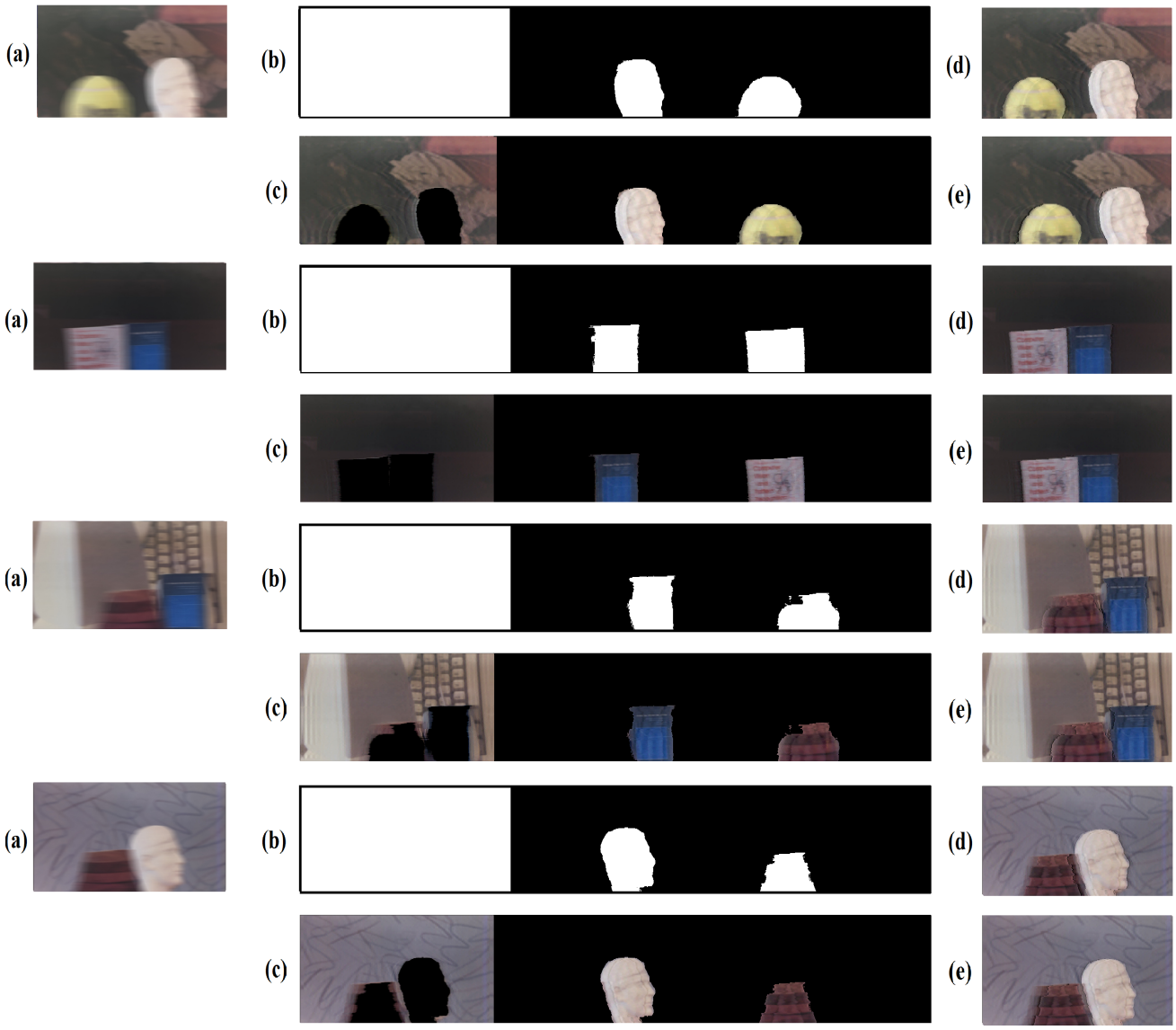


Fig. 22. (a) are real swiped images, which are made from three planes at different depths to form a diorama, constructed from cropped planes in [39], [40]. (b) are the estimated plane silhouettes, in descending depth order, with (c) being the recovered plane surfaces. From the recovered planes, we recover the EPIs of the scenes, and so can create novel views of the scene for different camera position, such as in (d) and (e).

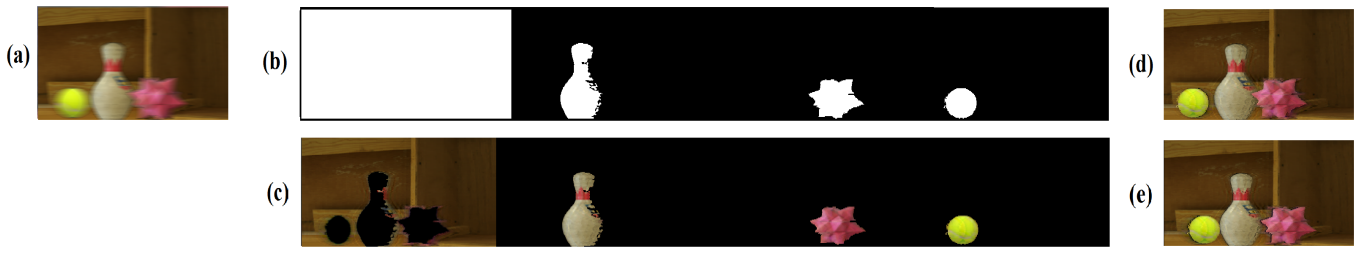


Fig. 23. (a) are simulated swiped images containing four planes, which are made from four planes at different depths to form a diorama, constructed from cropped planes in [39], [40]. (b) are the estimated plane silhouettes, in descending depth order, with (c) being the recovered plane surfaces. From the recovered planes, we recover the EPIs of the scenes, and so can create novel views of the scene for different camera position, such as in (d) and (e).

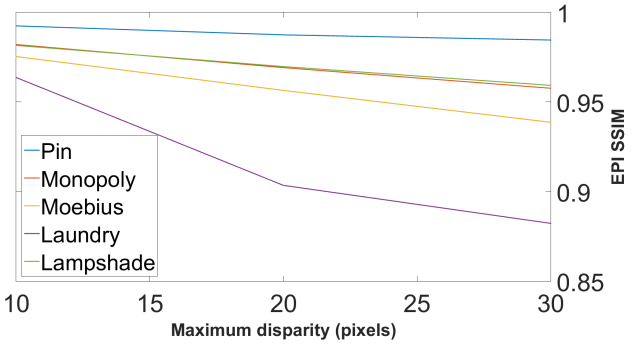


Fig. 24. A graph plotting the SSIM of the recovered EPI against the ground truth EPI as the maximum disparity in the swiped image is varied.

proposed technique of not recovering plane surfaces which are occluded during part of the swipe and introducing holes in the rendered image as the planes are shifted. From the comparison, it can be seen that the technique from this paper performs well against the blind deblurring algorithm, especially at the boundaries between planes, where there are more artifacts present in the deblurred images.

The images from the recovered EPI in Figure 25 are taken from the mid-point between x_{01} or x_{02}. The accuracy of the recovered EPI decreases as the position of the camera tends towards x_{01} or x_{02}, as the recovered pixels of occluded planes are less accurate if they are unoccluded for only a small proportion of the swiped image acquisition. However, Figure 26, taken from the top recovered EPI in Figure 25, shows the effect on the recovered EPI quality is small.

A further comparison is made on real swiped images in Figure 27 between the proposed method and a commercially available deblurring software package [44], using blind deconvolution for deblurring. The real swiped images in Figure 27(a) are deblurred using [44] in Figure 27(c) and compared with a rendered image from the recovered EPI using the technique from this paper, as shown in Figure 27(b). The comparison shows that a single image from a real recovered EPI performs well against [44], with improved definition of the outline of deblurred objects.

VI. Conclusion

In conclusion, we have presented a method to retrieve the EPI from a single swiped image. The method is derived from a depth-based layer model of the scene and a variation of the level set method. Numerical results on real and simulated images confirm the validity of the algorithm.

References

[1] M. Lawson, M. Brookes, and P. L. Dragotti, "Identifying a multiple plane plenoptic function from a swiped image," in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 1423–1427.

[2] ——, "Capturing the plenoptic function in a swipe," in *SPIE Optical Engineering + Applications*, San Diego, 2016, pp. 99 710O–99 710O.

[3] E. H. Adelson and J. R. Bergen, *The plenoptic function and the elements of early vision*. Vision and Modeling Group, Media Laboratory, Massachusetts Institute of Technology, 1991.

[4] C. Gilliam, P. L. Dragotti, and M. Brookes, "On the spectrum of the plenoptic function," *IEEE Transactions on Image Processing*, vol. 23, no. 2, pp. 502–516, 2014.

[5] J.-X. Chai, X. Tong, S.-C. Chan, and H.-Y. Shum, "Plenoptic sampling," in *Proc. Conf. on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., 2000, pp. 307–318.

[6] A. L. Da Cunha, M. N. Do, and M. Vetterli, "On the information rates of the plenoptic function," *IEEE Transactions on Information Theory*, vol. 56, no. 3, pp. 1306–1321, 2010.

[7] M. N. Do, D. Marchand-Maillet, and M. Vetterli, "On the bandwidth of the plenoptic function," *IEEE Transactions on Image Processing*, vol. 21, no. 2, pp. 708–717, 2012.

[8] L. McMillan and G. Bishop, "Plenoptic modeling: An image-based rendering system," in *Proceedings of the conference on Computer graphics and interactive techniques*. ACM, 1995, pp. 39–46.

[9] E. H. Adelson and J. Y. A. Wang, "Single lens stereo with a plenoptic camera," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2, pp. 99–106, Feb. 1992.

[10] R. Ng, M. Levoy, M. Brédif, G. Duval, M. Horowitz, and P. Hanrahan, "Light field photography with a hand-held plenoptic camera," *Computer Science Technical Report CSTR*, vol. 2, no. 11, pp. 1–11, 2005.

[11] M. Levoy, R. Ng, A. Adams, M. Footer, and M. Horowitz, "Light field microscopy," *ACM Transactions on Graphics (TOG)*, vol. 25, no. 3, pp. 924–934, 2006.

[12] R. Ng, "Fourier slice photography," in *ACM Transactions on Graphics (TOG)*, vol. 24, no. 3. ACM, 2005, pp. 735–744.

[13] R. C. Bolles, H. H. Baker, and D. H. Marimont, "Epipolar-plane image analysis: An approach to determining structure from motion," *International Journal of Computer Vision*, vol. 1, no. 1, pp. 7–55, 1987.

[14] H. Seok Lee and K. Mu Lee, "Dense 3d reconstruction from severely blurred images using a single moving camera," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 273–280.

[15] P. Chandramouli and A. Rajagopalan, "Inferring image transformation and structure from motion-blurred images," in *Proc. Brit. Mach. Vis. Conf.*, 2010, pp. 73–1.

[16] J. Berent and P. L. Dragotti, "Plenoptic manifolds," *IEEE Signal Processing Magazine*, vol. 24, no. 6, pp. 34–44, 2007.

[17] A. Gelman, P. L. Dragotti, and V. Velisavljevic, "Multiview image coding using depth layers and an optimized bit allocation," *IEEE Transactions on Image Processing*, vol. 21, no. 9, pp. 4092–4105, 2012.

[18] A. Gelman, J. Berent, and P. L. Dragotti, "Layer-based sparse representation of multiview images," *EURASIP Journal on Advances in Signal Processing*, vol. 2012, no. 1, pp. 1–15, 2012.

[19] J. Pearson, M. Brookes, and P. L. Dragotti, "Plenoptic layer-based modeling for image based rendering," *IEEE Transactions on Image Processing*, vol. 22, no. 9, pp. 3405–3419, 2013.

[20] R. J. Teather and I. S. MacKenzie, "Position vs. velocity control for tilt-based interaction," in *Proceedings of Graphics Interface 2014*. Canadian Information Processing Society, 2014, pp. 51–58.

[21] M. Levoy and P. Hanrahan, "Light field rendering," in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. ACM, 1996, pp. 31–42.

[22] H. Lin, C. Chen, S. Bing Kang, and J. Yu, "Depth recovery from light field using focal stack symmetry," in *Proc. IEEE Int. Conf. on Computer Vision*, 2015, pp. 3451–3459.

[23] A. Chakrabarti, T. Zickler, and W. T. Freeman, "Analyzing spatially-varying blur," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2010, pp. 2512–2519.

[24] S. Dai and Y. Wu, "Motion from blur," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2008, pp. 1–8.

[25] F. Couzinie-Devy, J. Sun, K. Alahari, and J. Ponce, "Learning to estimate and remove non-uniform image blur," in *IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2013, pp. 1075–1082.

[26] A. Levin, "Blind motion deblurring using image statistics," in *Advances in Neural Information Processing Systems*, 2007, pp. 841–848.

[27] D. L. Swets and J. J. Weng, "Using discriminant eigenfeatures for image retrieval," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 18, no. 8, pp. 831–836, 1996.

[28] C. Liu and H. Wechsler, "Gabor feature based classification using the enhanced fisher linear discriminant model for face recognition," *IEEE Transactions on Image processing*, vol. 11, no. 4, pp. 467–476, 2002.

[29] R. Johnson and D. Wichern, *Applied Multivariate Statistical Analysis*, ser. Applied Multivariate Statistical Analysis. Pearson Prentice Hall, 2007.

[30] S. Osher and J. A. Sethian, "Fronts propagating with curvature-dependent speed: algorithms based on Hamilton-Jacobi formulations," *Journal of computational physics*, vol. 79, no. 1, pp. 12–49, 1988.

(a) (b) (c) (d)

Fig. 25. Column (a) are simulated swiped images, which are made from two planes at different depths to form a diorama, constructed from cropped planes in [39], [40]. Column (b) are the newly rendered images at the camera mid-position from (a) using the techniques outlined in this paper. Column (c) are the ground truth images for comparison. Column (d) are the images from (a) deblurred using the software from [42], which implements the technique from [43].

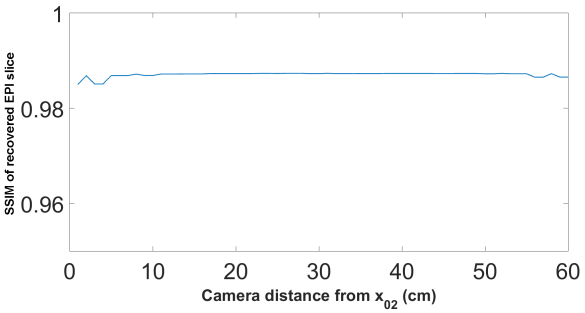
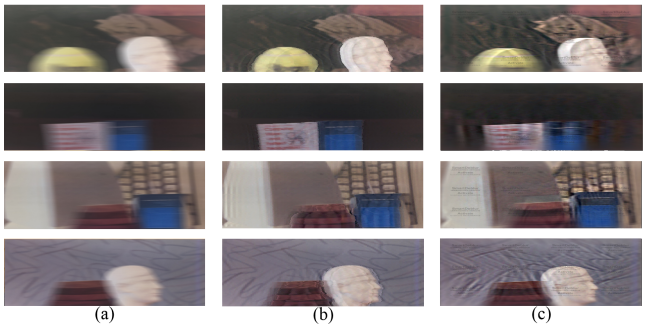


Fig. 26. A graph plotting SSIM of recovered EPI slices against camera position.



(a) (b) (c)

Fig. 27. Column (a) are real swiped images, of scenes comprising two planes at different depths to form a diorama, constructed from cropped planes in [39], [40]. Column (b) are the newly rendered images from the EPI estimated from (a) using the techniques outlined in this paper. Column (c) are the images from (a) deblurred using the software from [44].

[31] S. Osher and R. P. Fedkiw, "Level set methods: an overview and some recent results," *Journal of Computational Physics*, vol. 169, no. 2, pp. 463–502, 2001.

[32] H. Pitsch, "A consistent level set formulation for large-eddy simulation of premixed turbulent combustion," *Combustion and Flame*, vol. 143, no. 4, pp. 587–598, 2005.

[33] E. Maitre, T. Milcent, G.-H. Cottet, A. Raoult, and Y. Usson, "Applications of level set methods in computational biophysics," *Mathematical and Computer Modelling*, vol. 49, no. 11, pp. 2161–2169, 2009.

[34] R. Tsai, S. Osher *et al.*, "Review article: Level set methods and their applications in image science," *Communications in Mathematical Sciences*, vol. 1, no. 4, pp. 623–656, 2003.

[35] P. Smereka, "Semi-implicit level set methods for curvature and surface diffusion motion," *Journal of Scientific Computing*, vol. 19, no. 1, pp. 439–456, 2003.

[36] D. Adalsteinsson and J. A. Sethian, "A fast level set method for propagating interfaces," *Journal of Computational Physics*, vol. 118, no. 2, pp. 269–277, 1995.

[37] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM Journal on Imaging Sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[38] A. Chambolle, R. A. De Vore, N.-Y. Lee, and B. J. Lucier, "Nonlinear wavelet image processing: variational problems, compression, and noise removal through wavelet shrinkage," *IEEE Transactions on Image Processing*, vol. 7, no. 3, pp. 319–335, 1998.

[39] D. Scharstein and C. Pal, "Learning conditional random fields for stereo," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Jun. 2007, pp. 1–8.

[40] H. Hirschmuller and D. Scharstein, "Evaluation of cost functions for stereo matching," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Jun. 2007, pp. 1–8.

[41] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.

[42] J. JIA, "Robust deblurring software," http://www.cse.cuhk.edu.hk/ leojia/deblurring.htm, accessed: 2018-09-27.

[43] L. Xu, S. Zheng, and J. Jia, "Unnatural l0 Sparse representation for natural image deblurring," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Jun. 2013, pp. 1107–1114.

[44] SmartDeblur, "Smartdeblur deblurring software," http://smartdeblur.net/index.html, accessed: 2019-03-11.