Accurate Real-time Occlusion for Mixed Reality

David R. Walton
University College London
david.walton.13@ucl.ac.uk

Anthony Steed
University College London
a.steed@ucl.ac.uk

Figure 1: Real image augmented with a virtual laptop, with no occlusion handling (left), occlusion handling based directly on depths from an RGBD camera (centre), and occlusion using the proposed approach (right). Note that the inaccurate occlusion in the centre image is due to noise in the depth map, and the camera's inability to recover depth values for the user's watch.

ABSTRACT

Properly handling occlusion between real and virtual objects is an important property for any mixed reality (MR) system. Existing methods have typically required known geometry of the real objects in the scene, either specified manually, or reconstructed using a dense mapping algorithm. This limits the situations in which they can be applied. Modern RGBD cameras are cheap and widely available, but the depth information they provide is typically too noisy and incomplete to use directly to provide quality results.

In this paper, a method is proposed which makes use of both the colour and depth information provided by an RGBD camera to provide improved occlusion. This method, Cost Volume Filtering Occlusion, is capable of running in real time, and can also handle occlusion of virtual objects by dynamic, moving objects - such as the user's hands. The method operates on individual RGBD frames as they arrive, meaning it can function immediately in unknown environments, and respond appropriately to sudden changes. The accuracy of the presented method is quantified using a novel approach capable of comparing the results of algorithms such as this to dense SLAM-based approaches. The proposed approach is shown to be capable of producing superior results to both previous image-based approaches and dense RGBD reconstruction, at lower computational cost.

CCS CONCEPTS

• **Computing methodologies** **Visibility**; **Image processing**; **Mixed / augmented reality**;

VRST '17, November 8–10, 2017, Gothenburg, Sweden

ACM ISBN 978-1-4503-5548-3/17/11.
https://doi.org/10.1145/3139131.3139153

KEYWORDS

Mixed and Augmented Reality, User Interfaces, Image Processing

ACM Reference Format:
David R. Walton and Anthony Steed. 2017. Accurate Real-time Occlusion for Mixed Reality. In *Proceedings of VRST '17*. ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3139131.3139153

1 INTRODUCTION

Occlusion is a valuable depth cue, and an important tool to aid users of Mixed Reality (MR) in understanding the spatial relationships between real and virtual objects in the scene [Ellis and Menges 1998; Kalkofen et al. 2007]. In order for the virtual content in an MR application to appear consistent with the real content, it can be vital that occlusion between virtual and real objects is accurately rendered. Where MR is used as an aid to executing real-world tasks, such as navigating unknown environments [Livingston et al. 2003; Tsuda et al. 2006] using appliances [Feiner et al. 1993] or carrying out surgery [Kalkofen et al. 2007; Lerotic et al. 2007], it can be critical to ensure that content is visible to the user at the correct time. In such settings it may be unacceptable, and even pose a safety risk, for virtual content to inappropriately occlude real content in the user's view.

Occlusion between (opaque) real and virtual objects can be reproduced using the per-pixel depth of the real objects, using a technique similar to z-buffering. These depths could be obtained by rendering a geometric model of the scene, in which case the quality of the results will depend upon the accuracy and detail present in this geometric model, as well as the accuracy of the camera tracking.

Unfortunately, scene models are not usually readily available. For example, an MR mobile application would not have access to detailed geometric models of users' homes. It would be possible to use a real-time dense SLAM algorithm such as Kinect Fusion [Izadi et al. 2011], but this would come at a high computational

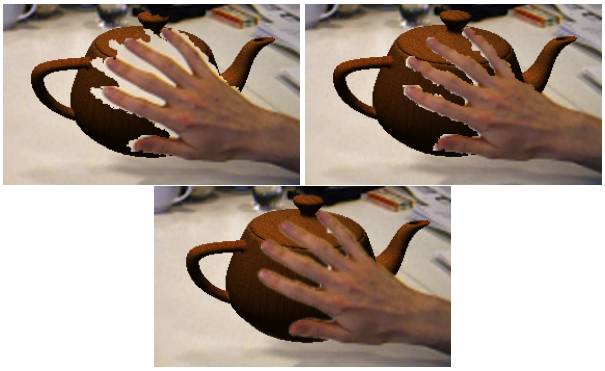

Figure 2: User's hand occluding a virtual teapot, in an MR scene. Top: Results using naïve approaches. Top Left: Unknown pixels assumed in front of virtual object. Top Right: Unknown pixels assumed behind virtual object. Right: improved result using the presented method. Images produced using the application detailed in sect. 5.3.

cost, require some time to produce an accurate reconstruction, and assume a static, rigid scene. In changing environments or when operating on resource-constrained devices, another solution for determining occlusion would be required.

RGBD cameras, which provide per-pixel colour and depth information, have become increasingly widely available in recent years. However, they suffer from temporal noise, and are frequently unable to determine accurate depth values for certain pixels. This can result in poor-quality results when these depth values are used to render occlusion in MR, as can be seen at the top of figure 2.

This paper presents Cost Volume Filtering Occlusion (CVF Occlusion), a method to use this noisy, incomplete data to accurately simulate the occlusion of virtual content by real content in video see-through augmented reality scenes. The method is capable of running in real-time on a laptop computer. It is also capable of detecting partial occlusion at a pixel, allowing it to avoid aliasing along occlusion edges.

The method makes use of an approach based upon cost volume filtering [Hosni et al. 2013], making use of the efficient guided filter [He et al. 2013]. A result produced using this method can be seen at the bottom of figure 2. Our work builds upon earlier literature applying cost volume filtering to mixed reality occlusion, adapting it to this problem domain. An efficient GPU implementation of the guided filter helps to enable this method to function in real time. In contrast to previous approaches, the per-frame runtime of the method is predictable and independent of the size of the occluded region to process.

Finally, a method to quantitatively assess the quality of this method is presented. This method is capable of assessing both the accuracy and the temporal stability of the output. This is used to demonstrate that the presented method offers competitive quality to real-time Simultaneous Localisation and Mapping (SLAM) algorithms at lower computational complexity, in addition to being able to handle occlusion by moving or deformable objects, such as a person's hands.

To summarise, the contributions of this paper are as follows:

- An efficient method to perform real-time occlusion culling in MR using cost volume filtering with a guided filter. CVF Occlusion offers similar quality to full 3D reconstruction approaches, at lower computational cost, and can handle dynamic scenes.
- An approach for quantitatively measuring the accuracy of MR occlusion algorithms which involve scene reconstruction and refinement over time.

2 RELATED WORK

It can prove challenging to correctly reproduce occlusion in a MR application. Whilst mutual occlusion between virtual content can be efficiently rendered in modern graphics hardware, handling occlusion between virtual and real content is non-trivial. This is because it is a priori unclear where the real content should occlude the virtual content, as the location of the real content is often not known. A number of authors have attempted to tackle this problem in the past, and some of the suggested approaches are summarised here.

2.1 Model-based Approaches

A number of well-established approaches exist for accurately rendering the occlusion of virtual objects by real objects, using a detailed geometric model of the real scene [Breen et al. 1996]. Provided accurate registration is possible, these approaches can render detailed occlusion between virtual and real objects very efficiently, by making use of GPUs.

Such techniques are only applicable to static or precisely tracked real objects, for which detailed models are available. Consequently, such methods are not capable of resolving occlusion in the common case where the user of an MR application moves their hand in front of a virtual object. Additionally, should tracking become inaccurate, virtual content will exhibit incorrect occlusion.

2.2 Contour Tracking

Another class of approaches attempts to remove the requirement to accurately model the real scene, by tracking the silhouette of the occluding real object(s) [Berger 1997; Klein and Drummond 2004; Lepetit and Berger 2000]. Such methods typically track these object edges using a variant of the active contours (snakes) algorithm [Kass et al. 1988]. Similar techniques can also be used to refine the occlusion estimates produced by model-based methods [Klein and Drummond 2004].

Although such approaches have the advantage of not requiring models of the real scene, they typically require sufficient processing power to track the silhouette contours. The tracked contours may also not be entirely accurate, either due to problems in fitting (e.g. local minima), or to insufficient resolution and/or flexibility of the contour model. The active contours algorithm requires initialisation, which may need to be performed by the user (as in [Lepetit and Berger 2000]); this may not always be desirable or practical.

These methods also implicitly make the assumption that occluders are opaque objects with continuous, definite boundaries of finite length. This assumption does not hold for a wide variety of real-world occluders. Examples include translucent objects such as

smoke or stained glass, as well as objects exhibiting subpixel-scale detail such as hair.

2.3 Masks and Mattes

The approaches discussed in sect. 2.1, 2.2 all attempt to solve the problem of determining where real-virtual occlusions occur in an MR application. Many [Berger 1997; Kanbara et al. 2000; Klein and Drummond 2004; Lepetit and Berger 2000; Ventura and Höllerer 2009] produce a binary label for each pixel in the augmented image, indicating whether a real-virtual occlusion occurs at that pixel, effectively segmenting the image. The binary image containing these labels, a "mask", can be used to composite the real and virtual images to produce the augmented image.

This is, however, not a complete solution to the problem, as some pixels in an image may be partially occupied by real content, and partly by occluded virtual content. Such pixels are common at the boundaries of objects, where their edges bisect pixels, but also occur due to translucent real objects, as well as objects on a subpixel scale (such as hair). Assigning binary labels to such pixels leads to a variety of errors, including aliasing along edges.

For this reason, in the related area of foreground-background separation (e.g. for image compositing), some methods instead attempt to develop a matte containing a scalar 'alpha' value in the range [0, 1] for each pixel [Smith and Blinn 1996]. Here, 1 indicates a pixel containing only foreground content, and 0 indicates a pixel containing only background content. Values in the (0, 1) range contain a combination of foreground and background. A high-quality matte can produce better results than a binary mask, avoiding problems such as aliasing along edges ("jaggies"), as well as properly handling small-scale and translucent content [Smith 1995].

CVF Occlusion also generates alpha mattes. In these mattes, 1 indicates a pixel where the virtual content is fully visible, and 0 indicates a pixel where it is fully occluded. Values in the (0, 1) range indicate partial occlusion.

2.4 Occlusion and Depth

Where per-pixel depth values are available, these may be used to determine occlusion between virtual and real objects. This may be achieved by simply comparing the depths - where a virtual object has a greater depth than a real object, it can be assumed that the real object occludes the virtual object at this pixel.

[Wloka and Anderson 1995] describe an MR system which makes use of this principle to correctly composite real and virtual images. In this application, the real depth measurements are acquired using a stereo camera pair and a stereo matching algorithm. These are then added to the z-buffering process used in rendering the virtual content, preventing virtual objects from being rendered at those pixels where real objects exist at lower depths. This allows for the real and virtual content to be efficiently composited, but may cause aliasing along occlusion edges, due to the lack of sub-pixel occlusion. [Kanbara et al. 2000] later developed an approach with improved depth estimation, and demonstrated its use on a head-mounted display (HMD).

A number of other papers have focussed on the similar problem of producing an alpha matte, separating a foreground object from its background. Performing this operation on a colour image alone is a challenging problem, and existing solutions typically require user input to specify the approximate object location, and have a high computational complexity, rendering them unsuitable for use in real-time applications [Rhemann et al. 2009; Wang and Cohen 2008].

The additional information provided by a depth map can be exploited to produce mattes efficiently, without requiring user input. Wang et al. [Wang et al. 2007] made use of depth information to automate the application of two established alpha matting techniques to video sequences. More recently, a number of approaches have been developed to perform video alpha matting in real-time [Wang et al. 2012, 2010; Zhu et al. 2009]. Applications of such algorithms include inserting participants in a videoconferencing system into a single virtual room, for more natural remote group chat [Lu et al. 2011]. [Crabb et al. 2008] proposed a method for real-time matting, based on a cross bilateral filter. Similarly to CVF Occlusion, this approach involves processing the depth and colour images using an edge-preserving filter. However, CVF Occlusion is designed for MR occlusion specifically, and uses a two-step filtering and thresholding approach to provide improved results.

2.5 Depth Map Enhancement

As mentioned in section 2.4, an accurate real depth map may be used to solve the real-virtual occlusion problem. However, the depth map can only be used to provide binary results at each pixel, which leads to some degree of aliasing along the occlusion edges.

A number of techniques exist to improve the quality of the depth map produced by an RGBD camera, using its colour output. One class of these approaches uses some variety of structure-transferring filter, such as a cross bilateral filter [Eisemann and Durand 2004], to move detail from the colour image to the depth map, and fill holes. L. Chen et al. [Chen et al. 2012] presented an approach which used morphological operators and suitably adapted cross bilateral filters to remove incorrect values, fill holes and remove noise from depth maps. A later approach by C. Chen et al. [Chen et al. 2013] instead finds the improved depth map as the minimum of an energy function using the input depths, estimated confidence values and the location of edges in the colour image.

Many works focus on the situation where a depth map and colour image are available, but the depth map is of lower resolution. Here, the task is to upsample and refine the depth map, to obtain per-pixel depth values for the larger colour image. Approaches have tackled this problem in a variety of ways, including cost volume filtering [Cho et al. 2013; Yang et al. 2007] and adapted bilateral filters [Chan et al. 2008]. [Du et al. 2016] used an edge-snapping approach, and also demonstrated how the output could be used to render improved MR occlusion.

2.6 Reconstruction and Occlusion

[Ventura and Höllerer 2009] demonstrated an MR system which compared a planar reconstruction of the environment to the observed colours to identify dynamic real-virtual occlusions. This approach works well under the assumption that the reconstruction is accurate and the scene is static. However, it implicitly assumes that dynamic objects are always located between the viewer and the virtual object, resulting in incorrect behaviour when (for example)

Table 1: Pixel category names and associated conditions.

'infront'	On virtual object, real depth < virtual depth
'behind'	On virtual object, real depth > virtual depth.
'process'	On virtual object, real depth unknown or equal to virtual depth
'ignore'	Off virtual object.

a user places their hand behind a virtual object, or the appearance of the scene changes. The method is also only capable of generating binary segmentations, rather than full alpha mattes.

A number of recent methods have been developed to produce dense, detailed 3D reconstructions of real scenes in real-time [Izadi et al. 2011]. This opens up the possibility of using such reconstructions to calculate occlusion, using a method similar to those described in sect. 2.1. This allows such techniques to be applied in situations where scene geometry is not known a priori.

Such methods are capable of producing high-quality results, but rely heavily on the quality of the reconstruction, as well as the accuracy of the camera tracking. The quality of the rendered occlusion will be lessened in situations where the reconstruction is not capable of accurately capturing the fine detail or transparency of a real object. Many techniques, including Kinect Fusion and InfiniTAM [Kahler et al. 2015] assume a static scene, and the occlusion will be very inaccurate if this assumption is violated (for example, if a user obscures a virtual object with their hand).

3 PRESENTED METHOD

3.1 Setup

The hardware system used by this method consists of a single RGBD camera observing a scene containing an MR marker. For the purposes of this paper, the authors made use of the ArUco library [Garrido-Jurado et al. 2014] to determine camera location, which tracks by using fiducial markers.

Although a marker-based tracking system was used here, alternative markerless tracking systems such as PTAM [Klein and Murray 2007] could also be used for this purpose. The camera location is only used to correctly position the virtual content in the rendering step. A marker-based system was chosen because stable, external tracking was needed to compare a range of occlusion techniques.

3.2 Rendering

The virtual content is rendered, using the transforms obtained from the tracking. In addition to rendering RGB pixels, the depth (i.e. z coordinate) of each pixel in camera space is also rendered, producing a virtual depth map. The minimum and maximum depths of each virtual object are recorded.

3.3 Pixel Categorisation

The virtual depth map and real depth map are compared, to categorise each pixel in the image. These pixel categories are used as input to the subsequent stages of the approach. The categories employed are listed in table 1.

The pixel categories are named according to whether the real scene is in front of or behind the virtual object at a given pixel.

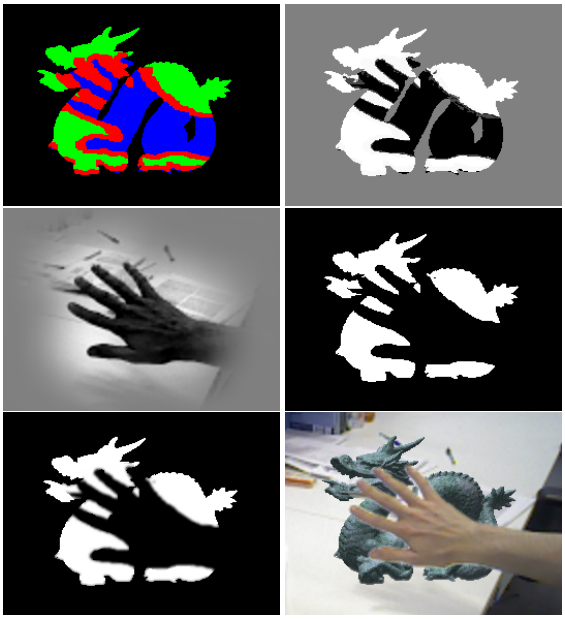

Figure 3: Stages involved in generating an augmented image using the proposed method. Top left: Categorisation applied to pixels. Classes are coloured as follows: 'infront', 'behind', 'process', 'ignore'. Top right: Initial costs. Middle left: Filtered costs. Middle right: thresholded costs. Lower left: Final matte. Lower right: Composited MR image.

As noted by [Nguyen et al. 2012], the style of depth camera used here suffers from lateral noise. This noise causes inaccurate depths at pixels near edges in the depth map. If the pixels were classified exactly as described in table 1, this would lead to some pixels erroneously being included in the 'infront' and 'behind' categories.

In order to mitigate this problem, a morphological (erosion) operator was used to remove pixels within a 3-pixel range of the border of the 'infront' and 'behind' categories. These pixels were included in the the 'process' category if occupied by the virtual object, or the 'ignore' category otherwise. As noted in [Nguyen et al. 2012], the lateral noise does not vary significantly with depth, so this 3-pixel range could be generally applied.

Note that the pixel categories generated here are similar in concept to the trimaps used in alpha matting techniques such as that of [Chuang et al. 2001]. The difference is that here, in addition to the three typical categories of foreground, background and unknown, we add a fourth 'ignore' category.

The top left image in figure 3 shows an example classification of pixels in an to which a virtual dragon is being added.

3.4 Initial Cost Calculation

Using these categories, per-pixel costs are generated, using a similar approach to that used by Hosni et al. for interactive image segmentation [Hosni et al. 2013]. In order to calculate these costs, colour models are first fitted to the 'infront' and 'behind' pixel categories, in a small area around the 'process' region. In this implementation,

we used histograms, however other colour models such as Mixture of Gaussians could also be used.

Once the models are fitted, the initial costs can then be generated. Each cost is a 32-bit floating-point value, and captures the likelihood of the real object being in front of or behind the virtual object at that pixel.

For pixels in the 'infront', 'behind' or 'ignore' categories, the cost is simply set to 1, 0 or 0.5 respectively. In the 'process' region, it is set to a value in the range $[0, 1]$ calculated as follows:

$$C(p) = \frac{P_{infront}(p)}{P_{infront}(p) + P_{behind}(p)}$$

Here, p is the pixel in question, and $P_{infront}$, P_{behind} are the probabilities of the pixel's colour under the respective histograms.

In many cases, these costs already resemble a reasonable image matte. Often, however, it contains inappropriate colour detail. This is removed in the subsequent cost filtering step.

An example of these initial costs can be seen in the top right of figure 3.

3.5 Cost Volume Filtering

Following this, a guided filter [He et al. 2013] is applied to the costs. The guided filter is an edge-preserving smoothing filter, which takes two images as input; a guidance image and an input image. The filter has the effect of smoothing the input image, whilst preserving hard edges present in the guidance image.

Here, the input image consists of the initial costs, and the guidance image is the colour image from the RGBD camera. The guided filter has the effect of smoothing the costs and removing some of the unnecessary detail, whilst preserving hard edges in the colour image. These hard edges typically correspond to occlusion boundaries.

This initial cost volume filter uses a fairly broad radius to provide a strong smoothing effect. After the filtering stage, the pixels in the 'process' region typically more closely resemble the desired matte, as can be seen in the middle left image of figure 3.

3.6 Matte Generation

Following the filtering stage, the filtered costs are thresholded to produce an initial matte for compositing the virtual object. Costs above 0.5 are set to 1.0, and those below 0.5 are set to zero. Only pixels from the 'process' region are used This produces an initial, binary matte.

A second guided filter is then applied to the matte. This filter uses a smaller radius, and has the effect of 'feathering' the occlusion edges. This removes the aliasing artefacts that would result from using the initial binary matte as-is. Results are particularly improved in cases where occluders have 'fuzzy' edges where many pixels should exhibit partial occlusion (for example the hair on a person's head, or a fluffy toy). Examples of thresholded costs and the final generated matte can be seen in the middle right and bottom left of figure 3, respectively. Here, one can see that the filter has removed the aliasing artefacts from the edges of the hand.

This process is adapted from the matting approach proposed by [Hosni et al. 2013]. However, Hosni et al. use an iterative approach, where the model fitting, filtering and thresholding stages are repeated multiple times. In our approach these stages are performed only once, improving the efficiency and allowing the process to run in real time. Since the depth map is already close to the true occlusion edges, a single iteration is typically sufficient. Future implementations could use multiple iterations if this is necessary. This would be useful if, for example, the RGBD sensor used has a lower-resolution depth sensor, or higher-resolution colour sensor than the one used here.

If multiple iterations are required, it is possible to take advantage of another property of the guided filter. When multiple images are to be filtered using the same guidance image and filter parameters, much of the computation can be reused to improve efficiency. This reduces the computational cost of subsequent iterations.

3.7 Compositing

Once an alpha matte has been produced, a compositing process is applied to produce the final augmented image.

This process is an approximation on the boundary of the occluding real object (i.e. at pixels where the matte value lies in the range $(0, 1)$). Here, the input colours at each pixel c_o initially contain a linear combination of the real background and foreground colours c_b and c_f:

$$c_o = \alpha c_b \cdot (1 - \alpha) c_f$$

In principle, the augmented image should have the colour c_a, where:

$$c_a := \alpha c_v \cdot (1 - \alpha) c_f$$

Here, c_v is the colour of the virtual object at this pixel. The approaches described here determine α, but not c_f. Consequently, the output image uses c_o in place of c_f. This could produce erroneous composites in certain situations, particularly when the foreground and background colours differ greatly at a pixel exhibiting partial occlusion. We found that this approximation produced reasonable results in our tests, however.

4 IMPLEMENTATION & EVALUATION METHOD

4.1 Implementation

This implementation of CVF Occlusion used a combination of CPU and GPU processing. The initial pixel categorisation, histogram fitting and cost calculation took place on the CPU, and were implemented in C++. The costs were then transferred to the GPU, and all subsequent processing took place there. This meant that the most expensive component of the approach, the two guided filter applications, could take advantage of efficient GPU implementation. It also minimised the information which needed to be transferred between CPU and GPU at each frame. The GPU code was written in OpenGL, which was also used to render virtual content and compose the output image. Components such as the guided filter were implemented in compute shaders.

The guided filter implementation used the efficient formulation described in [He et al. 2013]. This consists of a series of box filters, in addition to a number of simple pixel-wise image operations,

such as adding, subtracting or multiplying two images. The box filters are implemented using summed area tables [Crow 1984] (later works often refer to these as integral images). Consequently, our implementation is $O(N)$, where N is the number of pixels in the input image. It is independent of the radius of the filter employed. The summed area tables were calculated using parallel GPU prefix sums along rows and columns of the image.

4.2 Quality Evaluation Method

In order to quantify the relative quality of the results, it was necessary to develop a procedure for obtaining ground truth occlusions, and metrics to compare this ground truth to the output of CVF occlusion.

A small environment in which to carry out the experiment was constructed, containing an occluding object and some background objects. The RGBD camera was mounted on the quick-release plate of a sturdy tripod and pointed towards the scene. An MR marker, attached to a stiff foamcore board was placed off to the side of the scene, visible to the camera. The MR scene consisted of a single virtual object, between the real foreground and background objects, arranged to be partly obscured by the foreground.

We also intended to compare the results of CVF occlusion to those of a dense, Kinect Fusion-style algorithm. Such algorithms are intended to construct a scene model of gradually increasing quality over time, and need to view objects from multiple angles to produce accurate models. In order to compare a dense SLAM method to the proposed approach during the reconstruction, the following steps were repeated:

(1) Detach the camera from the tripod.
(2) Move the camera around the scene for a period of time, allowing Kinect Fusion to reconstruct the scene.
(3) Reattach the camera to the tripod.
(4) Mark a series of frames for comparison, record the appearance, depth, mask of the virtual object.
(5) Move the green screen behind the object, and capture the ground truth frame.
(6) Remove the green screen.

The dense SLAM algorithm used for the comparison here was InfiniTAM [Kahler et al. 2015]. This builds upon the original Kinect Fusion approach, adding a volume hashing mechanism enabling it to store the scene model compactly and reconstruct larger areas.

RGBD cameras, such as the one used in this example, an Xtion Pro, typically suffer from some temporal noise, particularly in the depth output. Because of this, it was decided to compare a series of consecutive frames during each cycle, to determine if this temporal noise caused the error of any of the approaches to vary significantly (here, 4 frames were used).

In practice, it was found to be easier to losslessly record the footage from the RGBD camera, and then apply the algorithms afterwards. This approach was chosen as it would not have been possible to run all of the compared approaches simultaneously in real-time.

4.2.1 Per-frame Accuracy. The mattes obtained using the occlusion methods were compared to ground truth mattes, in order to assess their accuracy. Figure 4 shows the process of developing a ground truth object matte from a green screen image. First, a photo editing tool is used to produce a matte b) from the input green screen image a). The authors found that attempting to produce this matte without user intervention sometimes led to inaccurate results for more challenging objects. The pixels of this matte not occupied by the virtual object c) are then set to zero, resulting in an accurate ground truth matte d).

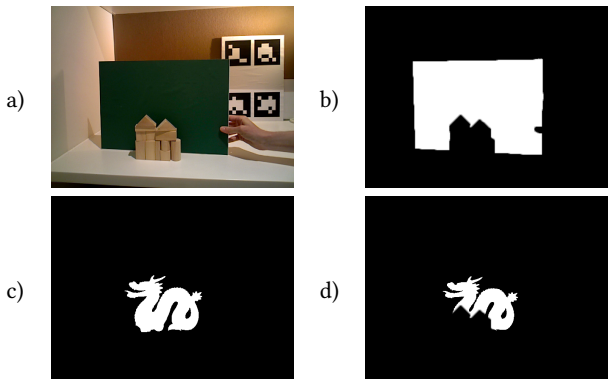

Figure 4: Developing a ground truth matte, using a green screen. a): Green screen placed behind foreground object. b): Matte extracted from greenscreen image. c): Pixels occupied by virtual object. d): Ground truth occlusion matte.

Once ground truth mattes M_G had been obtained, the accuracy of the automatically generated mattes M was then measured by calculating the mean squared error (MSE), as defined in equation 1.

$$MSE(M) := \frac{1}{|\Omega|} \sum_{p \in \Omega} (M(p) - M_G(p))^2 \quad (1)$$

4.2.2 Temporal Noise. In addition to improving the quality of mattes for individual frames, as measured by the MSE metric, it is also desirable for an occlusion method to remove the temporal noise present in the depth maps. This noise manifests as disturbing flickering artefacts along occlusion boundaries when the depth map is used directly.

In order to assess the ability of the occlusion methods to remove temporal noise, an additional metric was calculated. During step 4 of the capture procedure outlined in sect. 4.2, several frames are captured in sequence whilst the camera is fixed to a sturdy tripod. These frames contain static real and virtual scenes and so ideally the matte output would be constant over each of these sequences.

To measure the temporal noise in a sequence of output mattes, the variance at each pixel location in the matte is computed across the sequence. The mean of these variances is then taken, to provide a scalar measure of the temporal noise. As noted above, the ideal occlusion method would produce a sequence of identical mattes, resulting in a value of zero. A method that produced some degree of temporal noise would produce mattes with more variation, resulting in a higher (worse) value.

4.3 Compared Techniques

This evaluation method was applied to a number of different occlusion methods, to compare their performance.

Two naïve occlusion methods were implemented, which use the depth map from the RGBD camera directly. We refer to these as 'Direct1' and 'Direct2'. 'Direct1' assigns matte values m using just the real, unprocessed depth d_r and the virtual depth d_v at each pixel as follows:

$$m := \begin{cases} 1 & \text{if } d_r > d_v \\ 0 & \text{if } d_r \leq d_v \quad \text{or } d_r \text{ unknown} \end{cases}$$

'Direct2' is otherwise identical, but sets m to 1 where d_r is unknown. Thus, both methods compare real and virtual depths per-pixel to determine occlusion. In cases where the real depth is unknown, 'Direct1' assumes the virtual object is occluded, and 'Direct2' assumes it is not occluded. Missing values occur on both unoccluded and occluded parts of the virtual object, so neither of these approaches is inherently superior (although one might provide better results in a particular situation).

CVF occlusion was also compared to the earlier method of Crabb et al. [Crabb et al. 2008]. This approach was developed for foreground-background segmentation, rather than MR occlusion. Here it has been adapted by setting the threshold depth to the mean depth of the virtual object, and setting regions of the matte not located on the virtual object to zero. This adapted method is referred to as 'Crabb' in the results.

To provide context, the presented method was also compared to simpler approaches using other structure-transferring filters. In these approaches, an initial matte was generated using the pixel categorisation described in sect. 3.3, and the filter was then applied to refine it. The initial mattes were generated by setting pixel values from the behind category to 1, those from the unknown category to 0.5, and those from the infront or ignore categories to 0.

The filters tested were the guided filter used here, and an adapted joint bilateral filter. This joint bilateral filter was modified to only use information from pixels whose position relative to the virtual object is known (i.e. those from the 'infront' and 'behind' categories). In both cases, the filter was only applied to pixels in the 'process' category. These two methods are referred to as 'Guided' and 'Bilateral' in the results (according to the filter used).

CVF occlusion is referred to as 'CVF' in the results.

5 RESULTS

The experiment was repeated for a number of small MR scenes. These were constructed to be on a scale which allowed the camera to produce accurate depth values, and Kinect Fusion to reconstruct the scene accurately. Images of the scenes constructed are shown in figure 5.

The first scene, 'Normal' was designed to represent a typical use case, and the other two to represent more challenging situations. In 'Same Colour', the foreground and background are of a similar colour near the virtual object. In 'Specular', highly reflective foreground objects mean that the RGBD camera is often unable to obtain accurate foreground depths.

5.1 Per-frame Accuracy

Figure 6 shows the measured errors of the resulting mattes relative to the ground truth frames, using the MSE error metric. Each bar shows an average MSE for the frames captured consecutively in each cycle. In the first group, InfiniTAM has been exposed to the scene for a short length of time, with a stationary camera. In subsequent groups, the algorithm has been exposed to the scene from a wider range of viewing angles, for a longer length of time. The InfiniTAM error measurements (shown in orange) are the only ones which should show variation over time, as the other approaches do not make use of temporal information. Figure 7 shows some example results produced using CVF occlusion, with ground truth for comparison.

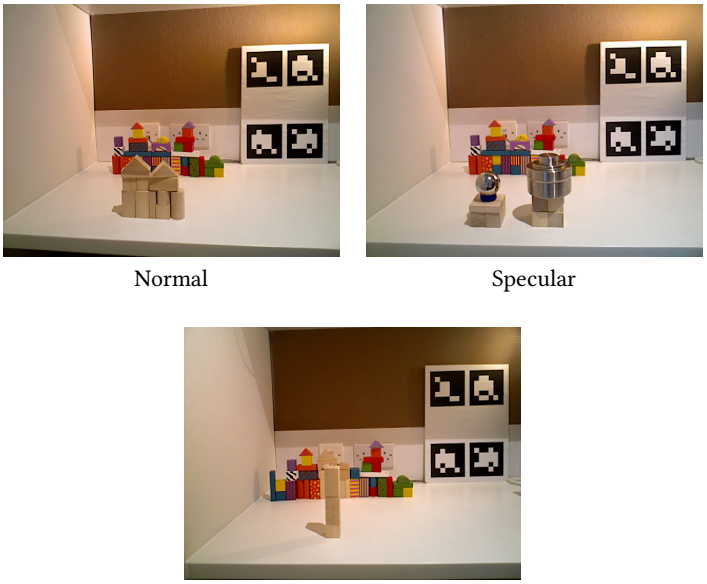

Figure 5: Colour frames from the three scenes constructed to test the output of the algorithm.

In the typical case 'Normal' scene, the proposed method provides better results than both direct approaches. The errors are also below those obtained using the other per-frame approaches.

Looking at the results of the two direct approaches, one can see that which is superior depends upon the situation. In the 'Normal' and 'Same Colour' sequences, where there are typically more unknown values behind the virtual object, Direct1 has lower errors. However, in the 'Specular' scene where there are many unknown values in front of the virtual object, Direct2 produces better results.

The accuracy of the CVF occlusion is lower in the two more challenging cases. In these cases the MSE is not always lower than both Direct1 and Direct2. However, in the 'Same Colour' example it is better than Direct1, and in the 'Specular' example it is better than Direct2, suggesting that the presented approach is more generally applicable.

The 'Specular' example is an interesting case where the approach of Crabb et al. often outperforms the presented approach in MSE. It should be noted, however, that Crabb suffered from much greater temporal noise in this sequence (see figure 8). Whether low temporal noise or MSE accuracy is a more important metric is likely to vary depending upon application domain.

Figure 6: Matte quality for each of the compared frames of the three RGBD sequences, as measured using Mean Square Error.

Figure 7: Example results obtained using CVF occlusion on each of the three test sequences (top row), and results using the corresponding ground truth mattes (bottom row).

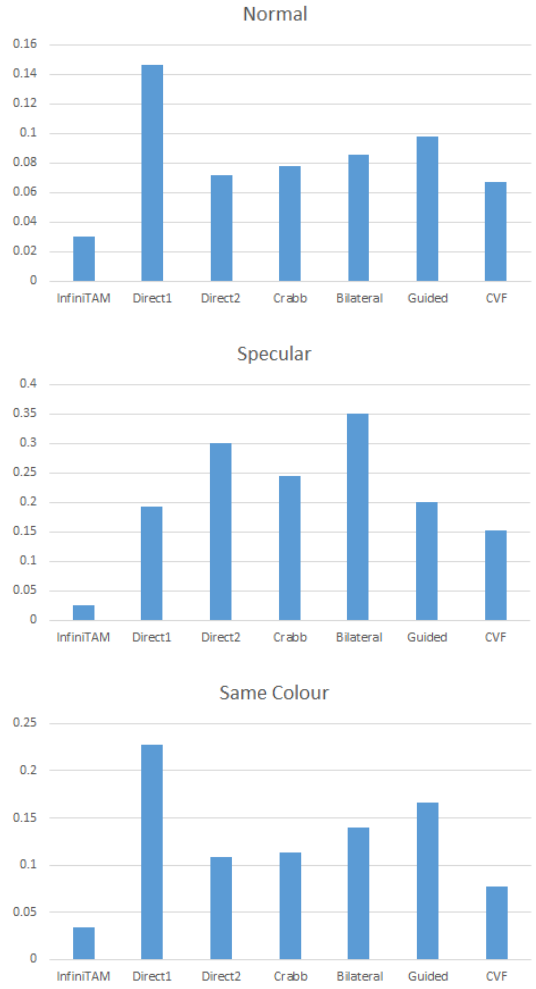

Figure 8: Temporal noise for each of the compared frames of the three RGBD sequences, measured as described in sect. 4.2.2.

The results from InfiniTAM gradually improve as the scene is exposed to the RGBD camera. For the first few frames in the 'Normal' and 'Same Colour' sequences, its error is much higher than the proposed approach. This highlights an advantage of using a per-frame approach; good results can be obtained, even on the first frame of a sequence.

It is also worth noting that, under certain conditions, such as rapid camera movement, reconstruction methods such as InfiniTAM can lose tracking, and this often results in the scene model becoming corrupted. Care was taken to avoid such rapid motions when capturing these sequences. The method presented here does not use any intra-frame state, and thus will continue to operate even when the camera image changes very quickly.

5.2 Temporal Noise

Figure 8 shows the average temporal noise estimates for each occlusion method, for each captured sequence. The noise estimates

Figure 9: Example results from the 'Same Colour' sequence, using the presented approach (left) and the baseline guided filter approach (right).

were calculated as described in sect. 4.2.2 and the average of these values was taken over each sequence.

As can be seen in figure 8, for the typical use case example 'Normal', the proposed approach offers a reduction in temporal noise. It is lower than both of the direct approaches, but also lower than Crabb et al., and the simplified guided filter and bilateral approaches. This shows the benefit of the two-step approach and thresholding stage, which reduce the direct dependence of the output matte on the noisy input depths.

In the more challenging 'Same Color' scene, although the MSE was not consistently below that of both direct approaches (as discussed above), the temporal noise of the proposed approach was still lower. In this sequence, CVF occlusion also showed significantly lower MSE than the baseline guided filter approach. The reason for this can be seen in figure 9. The thresholding stage used in CVF occlusion tends to produce results with well-defined boundaries, avoiding the errors which can be seen in the guided filter example.

InfiniTAM consistently offered the lowest temporal noise across all scenes, although this comes at the cost of its slow response to changes in real scene geometry (for example, if an occluding real object is moved).

5.3 Performance

In order to demonstrate the capability of CVF occlusion to work in real-time, a simple MR application was developed. This overlaid a single virtual object on the color image provided by the RGBD camera, using a fiducial marker for tracking. The occlusion method was applied at each frame, and the resulting matte used to compose the final MR image, which was displayed on the screen.

This application was tested using a computer with an Intel Core i7 processor and an Nvidia GTX 1080 GPU. An Asus Xtion Pro depth camera was used for both colour and depth input, at depth and colour resolutions of 640x480 pixels. Using this setup, the application was able to run consistently at 30 frames per second using CVF occlusion, the limiting factor being the framerate of the camera.

The occlusion method was also timed in isolation. In order to obtain meaningful values, it was necessary to forcefully synchronise the GPU (using `glFinish`). This means that the results may not reflect use of the method in a larger application, where the GPU could potentially run other tasks in parallel. They do provide a useful upper bound, however. With this synchronisation, each frame could be processed in between 5 and 7 milliseconds. Due to the $O(N)$ implementation of the guided filter, the processing time was largely independent of the kernel size used, and the proportion of pixels in the 'process' category, offering a consistent framerate.

The implementations of the Bilateral, Guided Filter and Crabb et al. approaches used here were implemented on the CPU, and able to run at near real-time (40-50ms per frame). More optimised GPU implementations of these approaches would likely have similar performance to CVF occlusion. Thus we believe that the cost of CVF occlusion is comparable with these other approaches, and it may be faster in many use cases.

6 DISCUSSION

CVF occlusion implicitly assumes a single virtual object - that is, that all real points lie either in front of or behind the virtual object. The same techniques could be applied to more complex scenes with alternating layers of virtual and real content, by processing each virtual layer separately. It would also be possible to process occlusion for each virtual layer in parallel.

The implementation of CVF occlusion used for the evaluation was not carefully optimised. It would be possible to further improve it to reduce the cost or allow implementation on lower-power mobile platforms. The summed area table calculation used in the guided filter in particular was implemented using two filter passes, and efficiency could be improved using the formulation proposed by [Nehab et al. 2011].

Occasionally, the depth camera will be unable to register any valid depth values for an occluding foreground object, due for example to small size, translucency or specularity. In these cases the object is only visible in the depth image as a patch of unknown values, surrounded by background pixels. In this case, the proposed approach will then incorrectly render the virtual object in front of the occluder. This problem is difficult to address using the information available to the algorithm, but was found to occur relatively rarely in practice.

7 CONCLUSION

CVF occlusion, an approach to producing occlusion mattes in real-time from the output of an RGBD camera was presented. This method operates independently on each frame, and is therefore capable of working immediately, and handling dynamic and deformable objects. A novel comparison approach was used to demonstrate that the method offered comparable results to a dense reconstruction approach (InfiniTAM), at lower computational cost.

The authors feel that these results demonstrate the potential for the use of image-based approaches coupled with off-the-shelf RGBD cameras to solve the MR occlusion problem, and show that full 3D scene reconstruction may not always be necessary. Alternatively, where reconstruction is employed for other purposes, it may be coupled with an image-based approach to refine the results, and/or handle occlusion by dynamic real objects.

Using information from preceding frames could help to further reduce the effects of temporal noise in the final output mattes, as well as allowing for better handling of cases where there are no valid depth values for a foreground object in a single frame (as discussed in sect. 6).

It would be interesting to explore the possibility of automatically detecting challenging scenarios, and adapting to them. For example,

in the 'Same Colour' sequence shown here, it would be possible to note that the background and foreground histograms are very similar, and adapt the subsequent stages of the approach.

This approach currently attempts to solve the occlusion problem by producing a matte, containing alpha values for each pixel. Strictly speaking, as mentioned in sect. 3.7, this is insufficient, and compositing also requires the real foreground colour to be extracted. Future methods might attempt to do so, in order to produce more accurate augmented images in the presence of translucency.

CVF occlusion could be a good fit for the GPUs available on mobile devices. A more optimised implementation of this approach could form part of an augmented reality system implemented on a mobile device, such as a Google Tango device, or a phone equipped with a stereo camera pair.

ACKNOWLEDGEMENTS

The authors would like to thank the UK's EPSRC and Imagination Technologies Limited for jointly funding this research, and Simon Julier and Tobias Ritschel for comments and suggestions.

REFERENCES

M-O Berger. 1997. Resolving occlusion in augmented reality: a contour based approach without 3D reconstruction. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*. IEEE, 91–96.

David E Breen, Ross T Whitaker, Eric Rose, and Mihran Tuceryan. 1996. Interactive occlusion and automatic object placement for augmented reality. In *Computer Graphics Forum*, Vol. 15. Wiley Online Library, 11–22.

Derek Chan, Hylke Buisman, Christian Theobalt, and Sebastian Thrun. 2008. A noise-aware filter for real-time depth upsampling. In *Workshop on Multi-camera and Multi-modal Sensor Fusion Algorithms and Applications-M2SFA2 2008*.

Chongyu Chen, Jianfei Cai, Jianmin Zheng, Tat-Jen Cham, and Guangming Shi. 2013. A color-guided, region-adaptive and depth-selective unified framework for Kinect depth recovery. In *Multimedia Signal Processing (MMSP), 2013 IEEE 15th International Workshop on*. IEEE, 007–012.

Li Chen, Hui Lin, and Shutao Li. 2012. Depth image enhancement for Kinect using region growing and bilateral filter. In *Pattern Recognition (ICPR), 2012 21st International Conference on*. IEEE, 3070–3073.

Ji-Ho Cho, Satoshi Ikehata, Hyunjin Yoo, Margrit Gelautz, and Kiyoharu Aizawa. 2013. Depth map up-sampling using cost-volume filtering. In *IVMSP Workshop, 2013 IEEE 11th*. IEEE, 1–4.

Yung-Yu Chuang, Brian Curless, David H Salesin, and Richard Szeliski. 2001. A bayesian approach to digital matting. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, Vol. 2. IEEE, II–II.

Ryan Crabb, Colin Tracey, Akshaya Puranik, and James Davis. 2008. Real-time foreground segmentation via range and color imaging. In *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08. IEEE Computer Society Conference on*. IEEE, 1–5.

Franklin C Crow. 1984. Summed-area tables for texture mapping. *ACM SIGGRAPH computer graphics* 18, 3 (1984), 207–212.

Chao Du, Yen-Lin Chen, Mao Ye, and Liu Ren. 2016. Edge Snapping-Based Depth Enhancement for Dynamic Occlusion Handling in Augmented Reality. In *Mixed and Augmented Reality (ISMAR), 2016 IEEE International Symposium on*. IEEE, 54–62.

Elmar Eisemann and Frédo Durand. 2004. Flash photography enhancement via intrinsic relighting. In *ACM transactions on graphics (TOG)*, Vol. 23. ACM, 673–678.

Stephen R. Ellis and Brian M. Menges. 1998. Localization of virtual objects in the near visual field. *Human factors* 40, 3 (09 1998), 415. http://search.proquest.com/docview/216459348?accountid=14511

Steven Feiner, Blair Macintyre, and Dorée Seligmann. 1993. Knowledge-based augmented reality. *Commun. ACM* 36, 7 (1993), 53–62.

S. Garrido-Jurado, R. Muaoz-Salinas, F.J. Madrid-Cuevas, and M.J. Maran-Jiminez. 2014. Automatic generation and detection of highly reliable fiducial markers under occlusion. *Pattern Recognition* 47, 6 (2014), 2280 – 2292. https://doi.org/10.1016/j.patcog.2014.01.005

Kaiming He, Jian Sun, and Xiaoou Tang. 2013. Guided image filtering. *IEEE transactions on pattern analysis and machine intelligence* 35, 6 (2013), 1397–1409.

Asmaa Hosni, Christoph Rhemann, Michael Bleyer, Carsten Rother, and Margrit Gelautz. 2013. Fast cost-volume filtering for visual correspondence and beyond. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35, 2 (2013), 504–511.

Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, et al. 2011. KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*. ACM, 559–568.

O. Kahler, V. A. Prisacariu, C. Y. Ren, X. Sun, P. H. S Torr, and D. W. Murray. 2015. Very High Frame Rate Volumetric Integration of Depth Images on Mobile Device. *IEEE Transactions on Visualization and Computer Graphics (Proceedings International Symposium on Mixed and Augmented Reality 2015* 22, 11 (2015).

Denis Kalkofen, Erick Mendez, and Dieter Schmalstieg. 2007. Interactive focus and context visualization for augmented reality. In *Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*. IEEE Computer Society, 1–10.

Masayuki Kanbara, Takashi Okuma, Haruo Takemura, and Naokazu Yokoya. 2000. A stereoscopic video see-through augmented reality system based on real-time vision-based registration. In *Virtual Reality, 2000. Proceedings. IEEE.* IEEE, 255–262.

Michael Kass, Andrew Witkin, and Demetri Terzopoulos. 1988. Snakes: Active contour models. *International journal of computer vision* 1, 4 (1988), 321–331.

Georg Klein and Tom Drummond. 2004. Sensor fusion and occlusion refinement for tablet-based AR. In *Mixed and Augmented Reality, 2004. ISMAR 2004. Third IEEE and ACM International Symposium on*. IEEE, 38–47.

Georg Klein and David Murray. 2007. Parallel Tracking and Mapping for Small AR Workspaces. In *Proc. Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07)*. Nara, Japan.

Vincent Lepetit and Marie-Odile Berger. 2000. A semi-automatic method for resolving occlusion in augmented reality. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, Vol. 2. IEEE, 225–230.

Mirna Lerotic, Adrian J Chung, George Mylonas, and Guang-Zhong Yang. 2007. Pq-space based non-photorealistic rendering for augmented reality. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2007*. Springer, 102–109.

Mark A Livingston, J Edward Swan II, Joseph L Gabbard, Tobias H Höllerer, Deborah Hix, Simon J Julier, Yohan Baillot, and Dennis Brown. 2003. Resolving multiple occluded layers in augmented reality. In *Proceedings of the 2nd IEEE/ACM International Symposium on Mixed and Augmented Reality*. IEEE Computer Society, 56.

Jiangbo Lu, Viet Anh Nguyen, Zeping Niu, Bhavdeep Singh, Zhiping Luo, and Minh N Do. 2011. CuteChat: a lightweight tele-immersive video chat system. In *Proceedings of the 19th ACM international conference on Multimedia*. ACM, 1309–1312.

Diego Nehab, André Maximo, Rodolfo S Lima, and Hugues Hoppe. 2011. GPU-efficient recursive filtering and summed-area tables. *ACM Transactions on Graphics (TOG)* 30, 6 (2011), 176.

Chuong V Nguyen, Shahram Izadi, and David Lovell. 2012. Modeling kinect sensor noise for improved 3d reconstruction and tracking. In *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2012 Second International Conference on*. IEEE, 524–530.

Christoph Rhemann, Carsten Rother, Jue Wang, Margrit Gelautz, Pushmeet Kohli, and Pamela Rott. 2009. A perceptually motivated online benchmark for image matting. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 1826–1833.

Alvy Ray Smith. 1995. Alpha and the history of digital compositing. *https://www.cs.princeton.edu/courses/archive/fall00/cs426/papers/smith95c.pdf* (1995).

Alvy Ray Smith and James F Blinn. 1996. Blue screen matting. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. ACM, 259–268.

Takahiro Tsuda, Haruyoshi Yamamoto, Yoshinari Kameda, and Yuichi Ohta. 2006. Visualization methods for outdoor see-through vision. *IEICE transactions on information and systems* 89, 6 (2006), 1781–1789.

Jonathan Ventura and Tobias Höllerer. 2009. Online environment model estimation for augmented reality. In *Mixed and Augmented Reality, 2009. ISMAR 2009. 8th IEEE International Symposium on*. IEEE, 103–106.

Jue Wang and Michael F Cohen. 2008. *Image and video matting: a survey*. Now Publishers Inc.

Liang Wang, Minglun Gong, Chenxi Zhang, Ruigang Yang, Cha Zhang, and Yee-Hong Yang. 2012. Automatic real-time video matting using time-of-flight camera and multichannel Poisson equations. *International journal of computer vision* 97, 1 (2012), 104–121.

Liang Wang, Chenxi Zhang, Ruigang Yang, and Cha Zhang. 2010. Tofcut: Towards robust real-time foreground extraction using a time-of-flight camera. In *Proc. of 3DPVT*.

Oliver Wang, Jonathan Finger, Qingxiong Yang, James Davis, and Ruigang Yang. 2007. Automatic natural video matting with depth. In *Computer Graphics and Applications, 2007. PG'07. 15th Pacific Conference on*. IEEE, 469–472.

Matthias M Wloka and Brian G Anderson. 1995. Resolving occlusion in augmented reality. In *Proceedings of the 1995 symposium on Interactive 3D graphics*. ACM, 5–12.

Qingxiong Yang, Ruigang Yang, James Davis, and David Nistér. 2007. Spatial-depth super resolution for range images. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*. IEEE, 1–8.

Jiejie Zhu, Miao Liao, Ruigang Yang, and Zhigeng Pan. 2009. Joint depth and alpha matte optimization via fusion of stereo and time-of-flight sensor. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 453–460.