Interactive Free-Viewpoint Video

Gregor Miller, Adrian Hilton and Jonathan Starck

Centre for Vision, Speech and Signal Processing,
University of Surrey, Guildford GU2 7XH, UK
{gregor.miller, a.hilton, j.starck}@surrey.ac.uk

Abstract

This paper introduces a representation for multiple view video to support high-quality interactive free-viewpoint rendering of moving people. Multiple wide baseline views are integrated into a view-dependent representation based on pair wise refinement of the visual-hull. An exact view-dependent visual-hull (VDVH) algorithm is presented to compute an initial approximation of the scene geometry. Refinement of the VDVH surface is then performed using photo-consistency and stereo. Interactive rendering with user control of viewpoint is achieved by front-to-back rendering of the refined surfaces with view-dependent texture. Results demonstrate that this representation achieves high-quality rendering of moving people from wide-baseline views without the visual-artefacts associated with previous visual-hull and photo-hull approaches.

Keywords: Free-viewpoint video, view interpolation, visual hull, wide baseline stereo

1 Introduction

Free-viewpoint video which allows rendering of real events from novel views is of interest for both broadcast production, video games and visual communication. Ultimately the goal is to allow the producer or user interactive control of viewpoint with a visual-quality comparable to captured video.

Multiple camera capture systems have been widely developed to allow capture or real events both in the studio and outdoor settings such as a sports arena. Switching between camera views has been used to produce effects such as freezing time in film or action replays in sports but requires closely spaced cameras to reduce the need for interpolation of intermediate views. Kanade et al. [14] pioneered the *Virtualized Reality*™ system with a 51 camera 5m hemi-spherical dome to capture and reconstruct moving people using narrow baseline stereo. Other researchers [12, 18, 17] have used the visual and photo-hull to reconstruct moving people from smaller numbers of widely spaced cameras. There has also been interest in real-time view-synthesis for video conferencing using photo-hull or stereo to correct viewpoint distortions [7, 1, 9]. These approaches reconstruct an approximate geometry which limits the visual quality of novel views due to incorrect correspondence between captured images [4].

To improve reconstruction quality research has addressed the spatio-temporal consistency of reconstruction [23, 11, 6]. Vedula et al. [23] introduced *scene flow*, an extension of optical flow to 3D volumetric reconstruction, to estimate temporal correspondence based on photo-consistency. Cheung et al. [6] introduced a new representation of visual-hull which directly incorporated colour for temporal matching to increase the effective number of cameras. Alternatively model-based approaches have been explored which reconstruct explicit representations of the scene [5, 20]. Due to the limited accuracy of geometric reconstruction and correspondence between views these approaches results in visual artefacts such as ghosting and blur. Loss of visual quality compared to captured video limits their application for visual content production.

To overcome these limitations recent approaches to novel view synthesis of dynamic scenes have used image-based rendering approaches with reconstruction of geometry only as an intermediate proxy for correspondence and rendering [21, 24]. Zitnick et al. [24] simultaneously estimate foreground/background segmentation and stereo correspondence. This system achieves highly realistic view synthesis but is restricted to a narrow baseline camera configuration (8 cameras over 30°). Starck et al. [21] introduced a view-dependent optimisation for high-quality rendering from wide-baseline views (7 cameras over 110°). This approach uses an initial coarse approximation of the scene geometry based on the visual-hull. The initial coarse approximation is iteratively optimised for stereo correspondence to render novel viewpoints. These approaches achieve a visual-quality comparable to the captured video but do not allow rendering of novel viewpoints at interactive rates.

In this paper we introduce a representation to allow high-quality free-viewpoint rendering of video sequences with interactive user control of the viewpoint. Expensive computation of correspondence between views is pre-computed and represented in a form which can be used for rendering novel views at interactive rates. Novel contributions of this work include: an exact view-dependent visual-hull algorithm to provide an initial coarse approximation of the scene; an image-based approach for efficient refinement of scene geometry to obtain accurate correspondence between views; and a multiple view scene representation which allows high-quality rendering at interactive rates without the visual artefacts of previous visual-hull and photo-hull approaches.

2 Exact View-Dependent visual-hull

In this section we introduce a method to efficiently compute the exact visual-hull surface visible for an arbitrary viewpoint from a set of silhouette images of a scene.

Laurentini introduced the visual-hull [16], a volume-based construct which completely encloses an object in a scene given a set of silhouette images. The visual-hull is a course approximation to the true geometry obtained by intersecting the camera silhouette cones. Other researchers [15, 19, 8] have used the visual and photo hull to reconstruct scenes from smaller numbers of widely spaced cameras. These approaches reconstruct an approximate geometry which limits the visual quality of novel views due to incorrect correspondence between captured images. Franco et al. [10] presented a method to recover the exact representation of the visual-hull corresponding to silhouettes with polygonal contours. Brand et al. [3] describe a technique where differential geometry is applied to obtain a close estimate to the exact visual-hull surface from silhouette contours. Image-based visual-hulls [17] used an approximation view-dependent visual-hull to render novel views without explicit reconstruction. Silhouette outlines for individual camera views are approximated by a piecewise linear polygonal representation. This approximation results in inexact computation of the view-dependent visual-hull introducing additional artefacts in the rendering of novel views.

Given a set of observed silhouette images of a scene the exact view-dependent visual-hull (VDVH) for an arbitrary viewpoint is the intersection of the camera rays with the visual-hull surface. In this section we show how the exact visual-hull geometry can be calculated efficiently for an arbitrary virtual camera view. Constraints on the ordering of silhouette intersections along the virtual image rays, together with projective invariants between intersections for different views, are used to efficiently evaluate the exact intersection with the visual-hull surface.

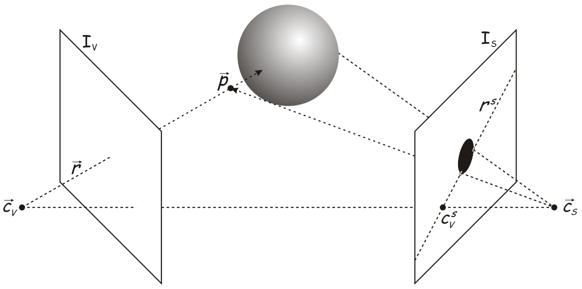
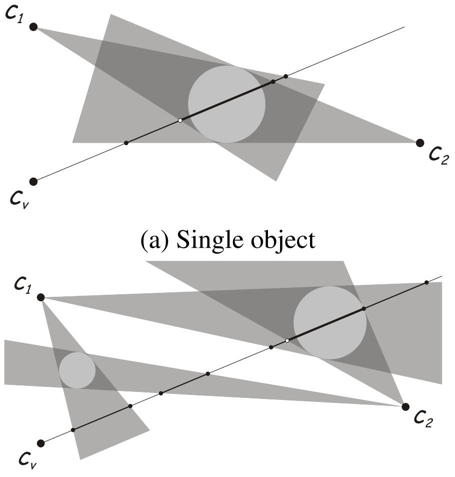


Figure 1: The ray $\vec{r}$ is cast from $\vec{c}_v$ and projected onto I_s to give r^s, the epipolar line. Rays are cast from $\vec{c}_s$ through the points of intersection between r^s and the silhouette boundary. These rays are triangulated with $\vec{r}$ to find the points on the visual-hull.



(a) Single object

(b) Two objects

Figure 2: Cross-section of the silhouette intersections along a virtual camera ray with centre of project $\vec{c}_v$ with silhouette images for two cameras with centre of projection $\vec{c}_1$ and $\vec{c}_2$. The first visible intersection point on the visual-hull surface marked an **o**.

2.1 Single View Visual-Hull Intersection

All methods of constructing the visual-hull find the intersection of the silhouette cones for each view, but differ in the approaches taken. For view-dependent visual-hull computation in this work the exact intersection of the silhouette cones is found in the image domain, taking advantage of multiple view geometry and projective invariants. In this section we first present the view-dependent visual-hull for a single view. Throughout this paper we use the notation, $\vec{x}$, to denote points or vectors in $\mathbb{R}^3$ and $x^j \in \mathbb{R}^2$ to denote the projection of a point or vector in the image plane of the j^{th} camera.

Consider the case of a single observed silhouette image I_s. The part of the visual-hull surface which is visible from a virtual view I_v is defined by the two-view projective geometry, as illustrated in Figure1. For a ray $\vec{r}$ from the virtual camera centre $\vec{c}_v$ through a pixel in I_v the intersection with the visual-hull surface is given by the intersection of the projection r^s of $\vec{r}$ in the observed view I_s. The projected ray is a two dimensional epipolar line in the image plane of I_s. All epipolar lines pass through the *epipole*, $c_v^s = P_s\vec{c}_v$, the projection of $\vec{c}_v$ onto I_s, where P_s is the camera projection matrix.

The intersection of r^s with the boundary of the silhouette returns an ordered set of intersection points, $\{u_k\}_{k=1}^K$, with corresponding intervals (u_k, u_{k+1}) on r^s inside and outside the silhouette. Here the intersections are assumed to be ordered along the epipolar line r^s starting from the epipole c_v^s with the epipole outside the silhouette. Whenever the epipole is inside the silhouette the next intersection on the epipolar line is removed. This follows from the assumption that all intersections must be in front of the camera (in other words objects are not behind the camera and the camera is not inside an object). There are two instersections with the silhouette for the example in Figure 1. In the case of a single image, for a silhouette intersection to correspond to a visible intersection of the ray $\vec{r}$ with the visual-hull surface it must satisfy the following condition:

> **Visual-Hull Visible Intersection Condition:** For a silhouette intersection, u_k, to correspond to an intersection of ray $\vec{r}$ with a visible part of the visual-hull surface the intersection number k along the epipolar line r^s must be odd.

This condition guarantees that the intersection point u_k is visible (the surface normal points towards the camera viewpoint). In the single-view case the exact intersection of the virtual camera ray $\vec{r}$ with the visual-hull surface is given by the 3D point $\vec{p}$ corresponding to the first intersection u_1 of the epipolar line r^s with the silhouette. This can be represented by the scalar distance d along the camera ray such that $\vec{p} = \vec{c}_v + d\vec{r}$. Given a point u on the epipolar line r^s there is a corresponding 3D point $\vec{p}(u)$ on the ray $\vec{r}$ with distance $d(u)$. The exact VDVH for a single view is given by the first silhouette intersection on the epipolar line of every ray through the virtual image I_v.

2.2 Multiple View Intersection Ordering

Given N views we have an ordered set of silhouette intersections $U^j = \{u_k^j\}_{k=1}^{K_j}$ for the epipolar line r^j of ray $\vec{r}$ projected onto the silhouette image I_s^j for the j^{th} view. Then the silhouette intersection from all views which corresponds to the intersection of ray $\vec{r}$ with the visual-hull is given by the following theorem:

> **Visual-Hull Intersection Theorem:** The silhouette intersection $u_k^j \in \{U^j\}_{j=1}^N$ which corresponds to the exact intersection of ray $\vec{r}$ with the visual-hull surface is the first silhouette intersection which satisfies the condition that for each of the views there is an odd number of silhouette intersections on the projection of ray $\vec{r}$ from the virtual camera centre $\vec{c}_v$ up to and including the point $\vec{p}(u_k^j)$.
>
> **Proof:** If there is an even number of intersections for any view j on the line segment between $\vec{c}_r$ and $\vec{p}(u_k^j)$ then for the j^{th} view the projection of $\vec{p}(u_k^j)$ is observed as outside the silhouette corresponding to empty space. Consequently if the projection of $\vec{p}(u_k^j)$ is not inside or on the silhouette for all views then it does not correspond to a point on the visual-hull. Therefore the *visual-hull visible intersection condition* must be satisfied for all views for intersection u_k^j to be on the visual-hull. This requires an odd number of silhouette intersections along the corresponding epipolar line r^j for all views. The first intersection for which this condition is satisfied will correspond to the visible intersection of ray $\vec{r}$ with the visual-hull surface.

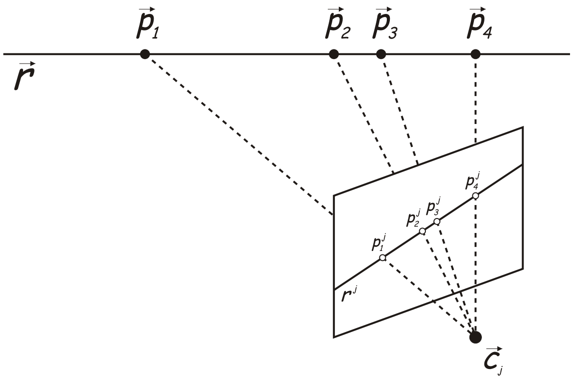


Figure 3: The cross ratio of $\vec{p}_{1-4}$ on $\vec{r} \in \mathbb{R}^3$ is equal to the cross ratio of $p_{1-4}^j = P_j\vec{p}_{1-4}$ on r^j in the j^{th} view.

This gives the exact intersection of the ray $\vec{r}$ with the visual-hull surface. Figure 2 illustrates the silhouette intersections for a virtual camera ray with two silhouette images in the case of single and multiple objects. In both cases the first visible intersection of the ray with the visual-hull surface is the first point which is inside the silhouette for both camera views. This is given by an odd number of silhouette intersections for each camera view as stated in the *Visual-Hull Intersection Theorem*.

2.3 Ordering by Projective Invariants

The theorem introduced in the previous section states that for a set of silhouette images the exact visual-hull intersection of a virtual camera ray $\vec{r}$ can be determined from the ordering of silhouette intersections for each view. In this section we show how projective invariants can be used to evaluate the relative ordering of silhouette intersections for different views without explicit computation of the 3D points $\vec{s}(u)$ along the ray. This allows computationally efficient evaluation of the exact intersection of each ray with the silhouette boundaries.

The *cross ratio* of four collinear points, p_{1-4}, is the only invariant in projective geometry [13], defined by:

$$x(p_{1-4}) = \frac{|\overrightarrow{p_1p_2}||\overrightarrow{p_3p_4}|}{|\overrightarrow{p_1p_3}||\overrightarrow{p_2p_4}|} \qquad (1)$$

where $\overrightarrow{p_k p_l} = p_l - p_k$. The cross ratio is constant across all domains for the same points, i.e. $x(p_{1-4}) = x(p^j_{1-4})$, as illustrated in Figure 3. This property can be used to compare the ordering of silhouette intersection points along the virtual camera ray $\vec{r}$ by comparison of their cross ratio along the epipolar lines for different views. To evaluate the cross ratio, three points are generated on the ray and projected onto all images. For example, $\vec{p}_1 = \vec{c_v} - 2\vec{r}$, $\vec{p}_2 = \vec{c_v} - \vec{r}$ and $\vec{p}_3 = \vec{c_v}$. These common points are projected onto view j to obtain three points on the epipolar line, r^j: p^j_1, p^j_2 and $p^j_3 = c^j_v$. The cross ratio x^j_k of the projected points with a silhouette intersection point $p^j_4 = u^j_k$ is calculated from Equation 1, and used to sort the points by increasing distance from the camera centre.

Ordering of silhouette intersections along the virtual camera ray $\vec{r}$, using the cross ratio x^j_k, is used to identify the silhouette intersection u^j_k which corresponds to the first visible intersection with the visual-hull surface. The corresponding 3D point on the visual-hull surface $\vec{s}(u^j_k)$ is reconstructed as the distance $d(u^j_k)$ along the ray from c_v. Repeating this process for virtual rays corresponding to each pixel in the virtual image, I_v, we obtain the exact view-dependent visual-hull as a distance image or z-buffer. Figure 4(a) illustrates the view-dependent visual-hull of a person for a 10 camera setup.

2.4 Computational Efficiency

There are various methods for constructing a visual-hull surface. The approach we have presented is an efficient algorithm for producing scene geometry. Our method adopts a *bin* structure to represent the silhouette boundary to increase the efficiency of intersection operations. The silhouette image I_s is divided into a number of bins, each one representing an angular range of epipolar lines. The silhouette boundary is split at each intersection with the edge of a bin, producing a number of small pixel lists indexed by the bin number. When performing epipolar line intersections the structure allows a fast look-up of the bin the epipolar line lies in, and the pixels on which to perform the intersections.

The complexity of our method is $O(sm^2 i)$ (assuming an image with $O(m^2)$ pixels, s the number of images and i the average number of intersections of a ray with the silhouette cone, $s \ll m$ and $i \ll m$), which is equivalent to the approximate solution of an image-based visual-hull [17]. Compared to a brute force volumetric approach ($O(n^3)$, n = *image size*) or a more efficient alternative ($O(n^2 \log n)$)[22] our technique has less complexity.

3 N-View Video Representation

Previous work has seen the visual-hull surface widely used for rendering novel viewpoints from multiple view video capture. The visual-hull is the maximal surface consistent with a set of silhouette images, and as such is only capable of approximating the true scene geometry. Errors arise in the form of extended surface volumes due to concavities and self-occlusion, or phantom volumes which result from the occlusion of multiple objects. The inaccuracies in the visual-hull produce erroneous correspondences between views, therefore rendering novel views based on its geometry will result in visual artefacts. This limits the quality of virtual views and prohibits their use in broadcast quality production.

We introduce a novel representation for interactive free-viewpoint rendering from wide-baseline multiple view video capture in this section. The initial process is an offline construction and refinement of view-dependent visual-hull (VDVH) surfaces. A multiple view video representation for online interactive rendering based on the refined surfaces is then presented.

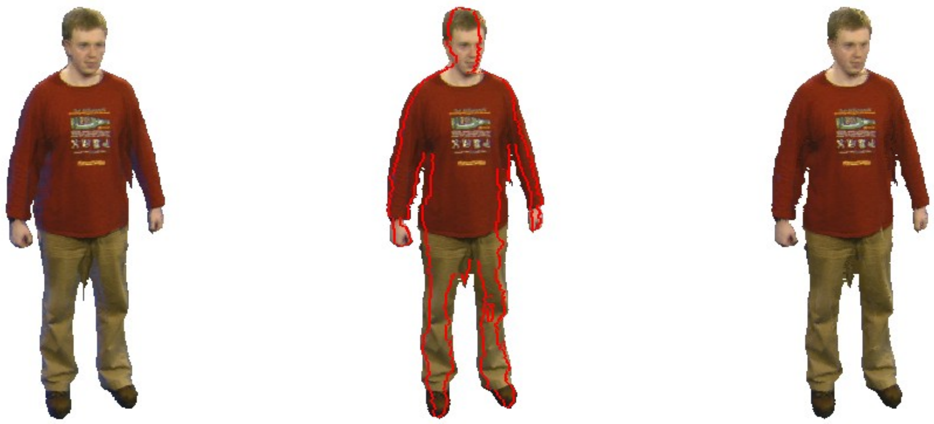
3.1 VDVH Refinement

The surface provided by the VDVH is an approximation which can be refined by applying a stereo matching algorithm. Direct computation of dense correspondence for wide-baseline views is an open problem in computer vision. Difficulties arise due to surface regions of uniform appearance, occlusion and camera calibration error. This work introduces an efficient image-based refinement of the VDVH using constrained stereo correspondence. Pixel-by-pixel refinement is applied in regions where colours from adjacent views are inconsistent. This process is demonstrated to achieve a surface approximation which allows novel viewpoint rendering with reduced visual artefacts from incorrect correspondence.

For every pair of physically adjacent cameras in the capture setup a VDVH is generated and refined. View-dependent refinement versus global reconstruction has been shown in previous work to improve rendering quality[21]. The reliability of correspondences is also improved in the presence of camera calibration error and changes in appearance with viewing direction.

A fixed virtual viewpoint is created halfway between two real cameras. The first step is to construct a depth map for this view from the VDVH. Textured meshes are generated from the VDVHs for the two real views adjacent to the virtual view. The meshes are projected onto the virtual view's image plane, and the overlapping areas of the projections compared. For every inconsistent pixel the depth at that pixel is refined.

The system is initialised by constructing the VDVH, V_p, for each real camera view $p \in [1, N]$ from the $(N-1)$ other views for all points inside the silhouette of the p^{th} view. A textured mesh is produced using the following process:



(a) View-dependent visual-hull (b) Overlapping area inside red line (c) Refined mesh representation

Figure 4: Stages in the refinement process at the mid-point between two cameras

1. Construct a step-discontinuity constrained triangulated mesh M_p from V_p by connecting adjacent pixels and applying distance threshold $t_d = 7\Delta x$, where Δx is the sampling resolution at the triangle vertex closest to the virtual camera centre.
2. Allocate each vertex of M_p a texture coordinate corresponding to its image pixel in I_p.

For each pair of adjacent cameras c_j and c_k with images I_j and I_k, $j,k \in [1,N]$. The refined representation M_{jk} is obtained as follows:

Refinement: Define the projection matrix P_{jk} of a virtual camera c_{jk} (positioned at the midpoint of the line connecting c_j and c_k) by copying the intrinsic parameters from a real camera and interpolating the extrinsic parameters of c_j and c_k. For this novel viewpoint:

(a) Evaluate the VDVH, V_{jk}, and corresponding depth map, D_{jk}, for the reference view from N real camera views.

(b) Render the reconstructed meshes M_j and M_k onto the virtual view to obtain images I^j_{jk} and I^k_{jk} for the parts of the mesh visible from the reference view.

(c) For each pixel u in the reference image which has colour in both I^j_{jk} and I^k_{jk}:

i. Test for colour consistency: $|I^j_{jk}(u) - I^k_{jk}(u)| < t_c$ where $I(u)$ is the RGB colour triplet for pixel u in image I and t_c is a threshold based on the camera noise. The colour distance is defined as the difference between the two normalised RGB vectors.

ii. If u is not colour consistent between images the depth map at u is refined using stereo matching. $D_{jk}(u)$ represents the distance along the virtual ray r from the camera centre c_{jk} to the visual-hull intersection. Refinement starts at this depth and is constrained to lie inside the visual-hull. An $m \times m$ window is used to evaluate the normalised cross-correlation between camera images I_j and I_k along the epipolar line for each view. The depth $d(u) = D_{jk}(u) + d'$ which gives the maximum correlation between views is taken as the refined depth estimate, or the original point is retained if no better match was found.

iii. The corresponding pixel in the depth map $D_{jk}(u)$ is updated with the refined depth estimate $d(u)$. The three-dimensional point at this depth is computed and projected into I_j and I_k to retrieve the RGB values.

Output: D_{jk} contains depths from non-overlapping, overlapping and refined regions. The refined reference mesh M_{jk} is constructed from D_{jk} with two colours per vertex for view-dependent rendering.

Stages of the refinement process are presented in Figure 4.

This algorithm constrains the refined surface for a camera pair to lie inside the visual-hull. The refined mesh is evaluated offline for each pair of adjacent cameras. This provides the basis for online rendering of novel views with a higher visual quality than that obtained with the visual-hull with wide-baseline views.

The border of the overlapping region is not refined since one of the cameras will have an unreliable view of the surface at these points. The colour threshold t_c is set to 0.05 for extensive surface refinement and 0.1 for conservative refinement. Throughout this work a 13×13 window is used in the stereo matching algorithm.

Occlusion in the reference view is not currently taken into account. For complex scenes there may be regions of the surface visible from both views which are not visible in the reference view. In the results presented for free-viewpoint rendering of individual people this has not been found to produce visible artefacts. However, in more complex scenes with multiple people a reference representation with multiple depths per pixel may be required.

3.2 Representation of N-View Video for Interactive Free-Viewpoint Rendering

For free-viewpoint rendering the scene is represented by the R refined surface meshes M_{jk} and view-dependent texture maps T_{jk} for all adjacent pairs of camera views. Rendering of novel views at interactive rates is achieved by rendering the set of R meshes in back-to-front order with back face culling enabled. The mesh generated from the camera furthest from the current viewpoint is rendered first, followed by the next furthest, and so on. The depth buffer is cleared between rendering each mesh to remove small artefacts from overlapping meshes (caused by errors in the visual-hull, from discretisation in the original images and camera calibration).

The ordered rendering of the refined meshes guarantees that each pixel u of the final novel view image I_v is rendered from the closest refined view containing a colour for u. All refined meshes are rendered to ensure that any missing surfaces which may occur due to occlusion are included in the final rendering.

View-dependent rendering of each refined mesh is performed by blending the texture from the captured images I_j and I_k according to the angle between the camera and rendered view point. As in previous view-dependent rendering[21] this ensures a smooth transition between views using the estimated correspondence. At the location of the camera viewpoints the rendered image is identical to the captured image.

View-dependent rendering of multiple refined meshes rather than a single refined visual-hull gives the best correspondence between the adjacent views. If stereo correspondence from all views are incorporated into a single representation then errors due to inconsistent correspondence and camera calibration occur. The use of a local refined representation ensures high quality rendering with accurate reproduction of surface detail.

3.3 Computation and Representation Cost

Representation of the scene requires R meshes and associated textures to be stored for each frame of the multiple view video sequence. The rendering cost is the total cost of rendering each of the individual meshes. If the camera image size is $P \times Q$ then each mesh has $O(PQ)$ vertices and the total cost of rendering is $O(RPQ)$. In the standard definition video used in this work $R = 8 - 14$, $P = 720$ and $Q = 576$ giving worst case representation and rendering cost of $6M$ textured triangles. In practice both the representation and rendering cost are an order of magnitude smaller as the foreground object only occupies a fraction (typically 25%) of the viewing area in any scene and approximately 50% of the triangles are back-facing for any given novel view. This gives representation cost at each frame of $1M$ triangles. Rendering can be achieved at interactive rates (greater than 25 frames per second) on consumer graphics hardware.

4 Results

In this section we present results and comparative evaluation for interactive free-viewpoint rendering of people. Capture was performed with two multiple camera studio configurations: the first setup comprises ten cameras equally spaced in a circle of radius 5m at a height of 2.5m looking towards the centre of the space, giving a baseline of 2.4m and capture volume of $3m^3$; the second setup comprises 8 cameras, seven in an arc of 110 degrees of radius 4m at a height of 2.5m, giving a baseline of 1.2m and capture volume $2.5m^3$ (the last camera gives a top-down view). Synchronised video sequences were captured at 25Hz PAL-resolution progressive scan with Sony DXC-9100P 3-CCD colour cameras. Intrinsic and extrinsic camera parameters were estimated using the public domain calibration toolbox [2]. Camera calibration gives a maximum reprojection error of 1.6 pixels (0.6rms) averaged across the cameras which is equivalent to a reconstruction error in the order of 10mm at the centre of the volume.

Rendering was implemented using OpenGL on a 3.2GHz P4 with 1GB RAM and a nVidia 6600GT graphics card. This implementation gives interactive rendering at 43 frames-per-second for novel viewpoints with the 8 camera setup and 34 frames-per-second for the 10 camera setup. Pre-computation with the 8 camera setup takes approximately 3 minutes per frame on a 3.2GHz PC.

Figure 5 shows novel rendered views of a person at the mid-point between each real camera view for the 10 camera setup. Results demonsrate the quality of rendered views which correctly reproduce detailed scene dynamics such as wrinkles in the clothing. This sequence also demonstrates that using a limited number of cameras high-quality novel view synthesis can be achieved for a complete circle surrounding the subject. Figure 6 shows frames from a rendered video sequence at two novel viewpoints with the camera viewpoint at mid-points between the real cameras. Figure 7 shows interactive free-viewpoint video rendering of novel views for a closely spaced sequence of views with the 8 camera setup. The rendering based on the refined representation reproduces hair and clothing movement. This representation eliminates visual

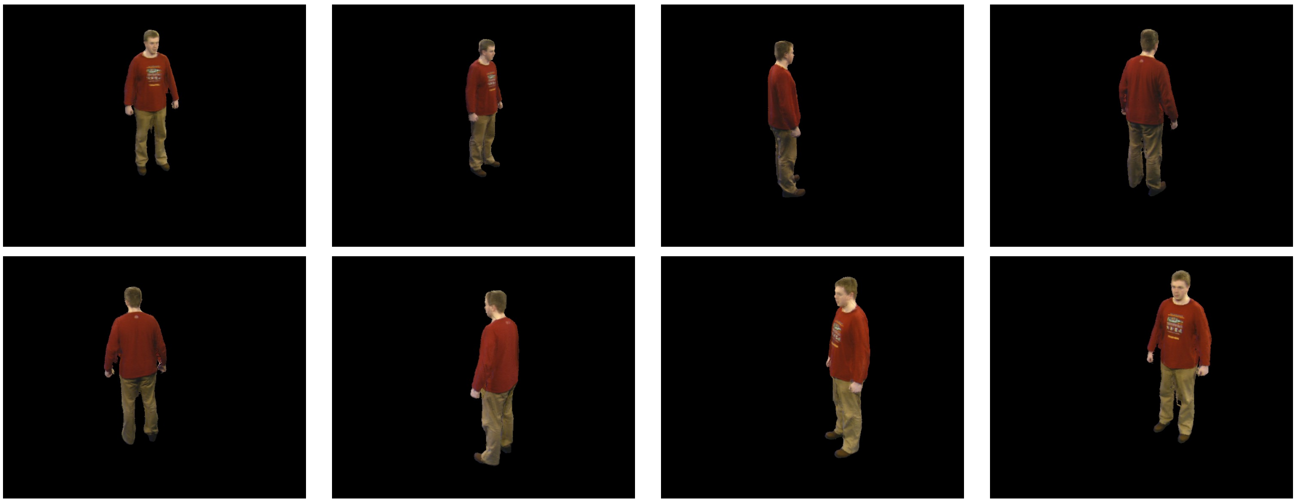


Figure 5: Free-viewpoint synthesis of a novel views around the person at the mid-points between the real camera views for a single time instant

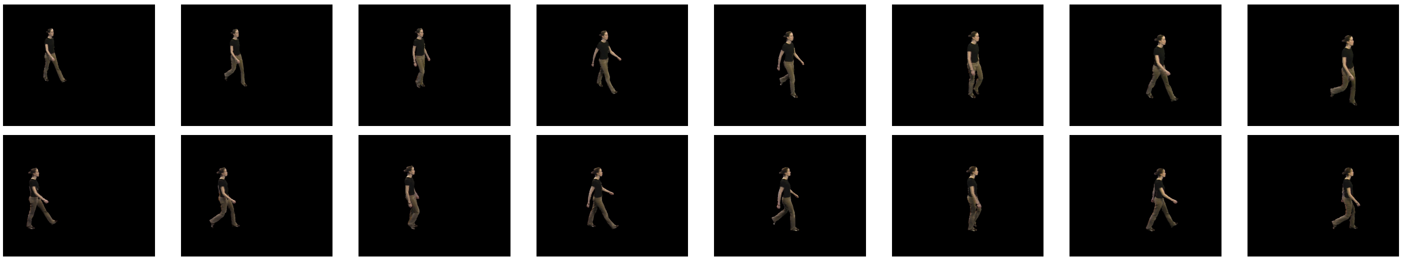


Figure 6: Video sequences for two novel views of a person

artefacts such as ghosting due to incorrect correspondence which occur with previous visual and photo-hull based free-viewpoint video techniques.

A comparative evaluation of free-viewpoint rendering quality from wide-baseline views has been performed comparing visual-hull and photo-hull and with the representation based on stereo refinement introduced in this work. Figure 8 presents comparative results for rendering of multiple video frames from a novel viewpoint rendering for a sequence captured with the 8 camera setup. This comparison, and that of the close-up shown in Figure 9, demonstrates that visual artefacts present in the visual-hull and photo-hull rendering due to incorrect correspondence between views are not visible in the refined stereo surface. The detailed pattern on the girl's dress is correctly reproduced demonstrating high-quality rendering with interactive viewpoint control. Furthermore as the proposed representation and refinement is pre-computed rendering is performed at above video-rate on standard graphics hardware.

5 Conclusions and Discussion

A representation for high-quality free-viewpoint rendering with interactive viewpoint control from multiple view wide-baseline video capture has been introduced. The representation is based on the pre-computation of stereo correspondence between adjacent wide-baseline views. Wide-baseline stereo correspondence is achieved by refinement of an initial scene approximation based on the view-dependent visual-hull (VDVH). A novel algorithm for efficient VDVH computation has been presented which evaluates an exact sampling of the visual-hull surface for a given viewpoint. To estimate wide-baseline correspondence the VDVH for the mid-point between adjacent views is refined based on photo-consistency and stereo correlation. This produces a refined representation of the visible surface geometry and appearance with accurate correspondence between views.

Interactive rendering of novel viewpoints is performed by back-to-front rendering or the refined representation starting from viewpoints furthest from the desired views and finishing with the closest viewpoint. Rendering is performed at video-rate (25Hz) on consumer graphics hardware allowing interactive

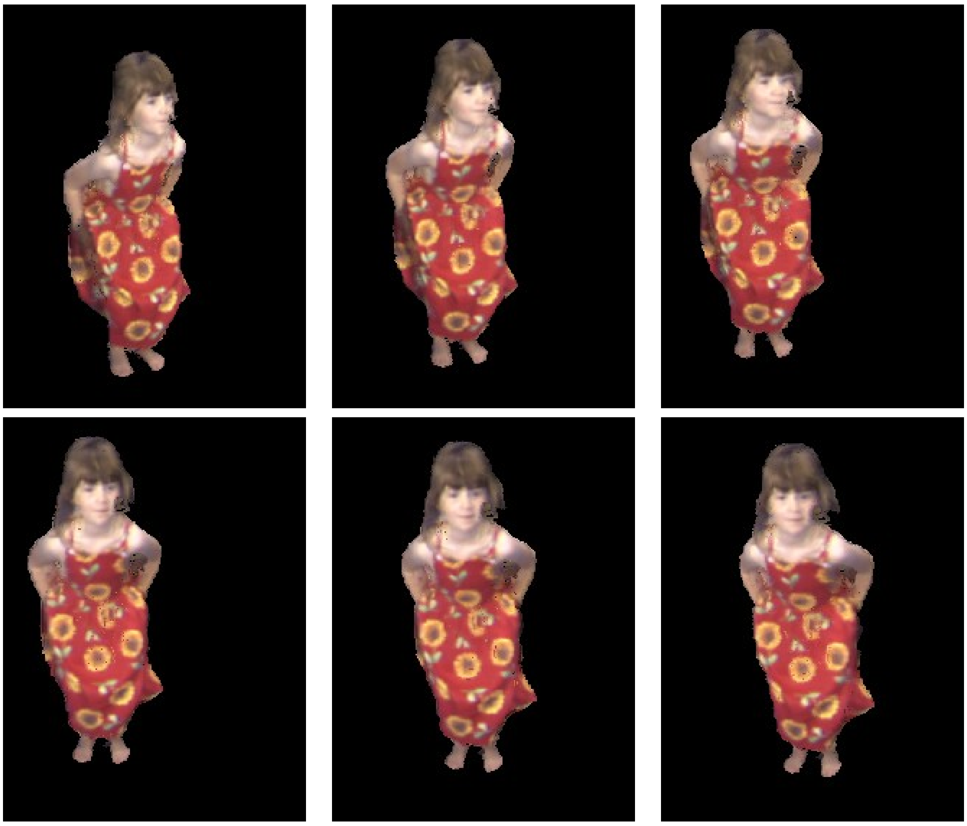


Figure 7: Video-rate interactive free-viewpoint synthesis at a single time instant

viewpoint control. Results from 8 and 10 camera multi-view wide-baseline studio capture demonstrate high-quality rendering of people with reduced visual artefacts. Comparative evaluation with previous visual and photo-hull approaches demonstrates that visual artefacts such as blur and ghosting are removed. The representation achieves high-quality rendering with accurate reproduction of the detailed dynamics of hair and clothing.

Two limitations of the present approach need to be addressed in further work. Errors in the silhouette segmentation result in artefacts at the border of the rendered person. Further work is required to optimise the boundary segmentation together with the surface refinement. Secondly, the representation currently assumes that the refined surface at the mid-point between views includes all overlapping visible surface regions for the adjacent views. This assumption in not guaranteed due to occlusion. In practice for sequences of individual people this has not been found to be a problem. To overcome this limitation for rendering more complex scenes a multiple layer representation could be used.

Acknowledgements

This research was supported by EPSRC grants GR/M88075 ‘Video-Based Representation of Dynamic Scenes’ and EPSRC GR/S27962 in collaboration with BBC R&D.

References

[1] H. Baker, D. Tanguay, I. Sobel, D. Gelb, M. Goss, B. Culbertson, and T. Malzbender. The coliseum immersive teleconferencing system. In *Proc.Int.Workshop on Immersive Telepresence*, 2002.

[2] J-Y Bouguet. Camera calibration toolbox for matlab: www.vision.caltech.edu/bouguetj/calib-doc. Technical report, MRL-INTEL, 2003.

[3] Matthew Brand, Kongbin Kang, and David B. Cooper. An algebraic solution to visual hull. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2004.

[4] C. Buehler, M. Bosse, L. McMillan, S. Gortler, and M. Cohen. Unstructured Lumigraph Rendering. In *SIGGRAPH*, pages 425—432, 2001.

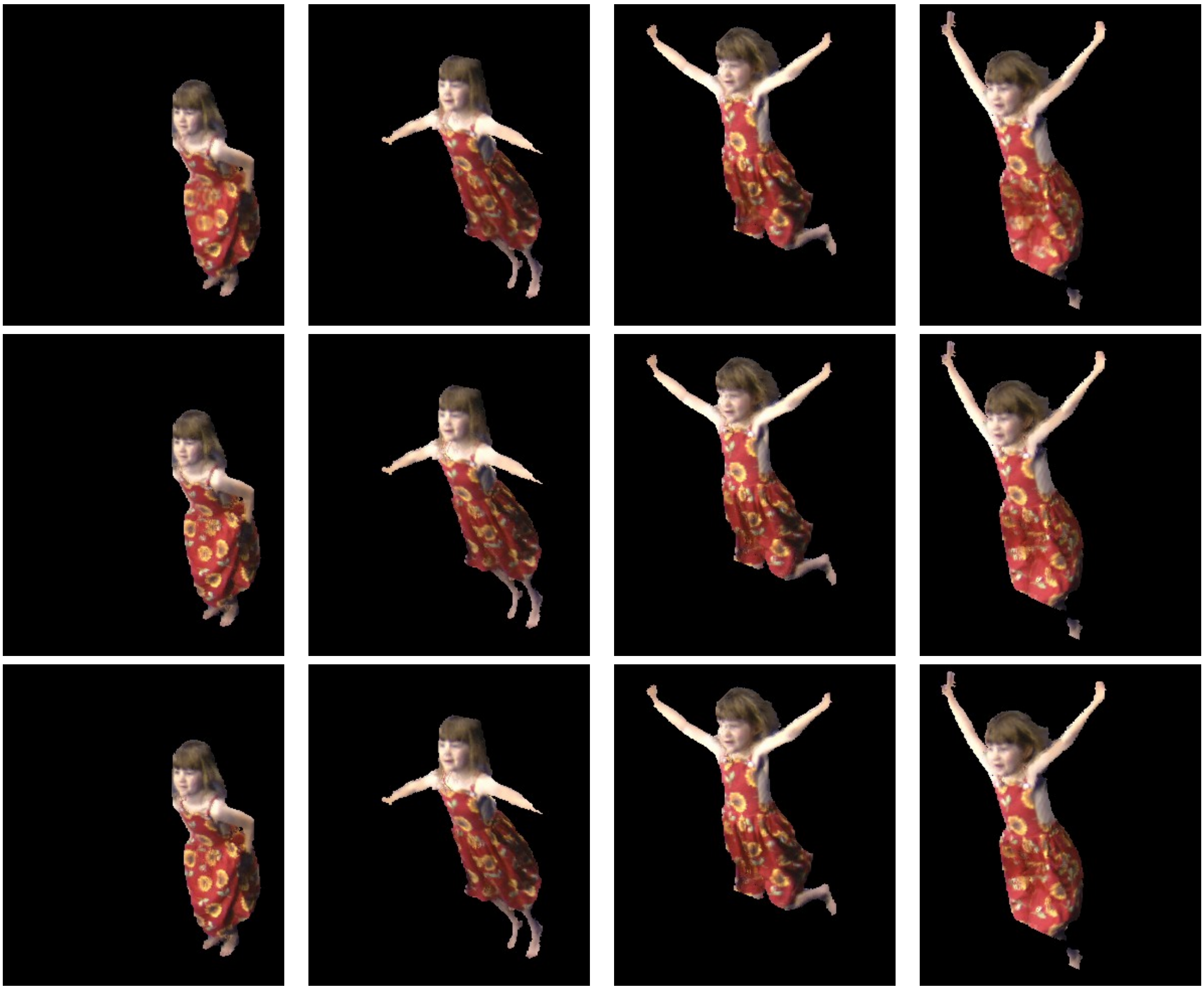
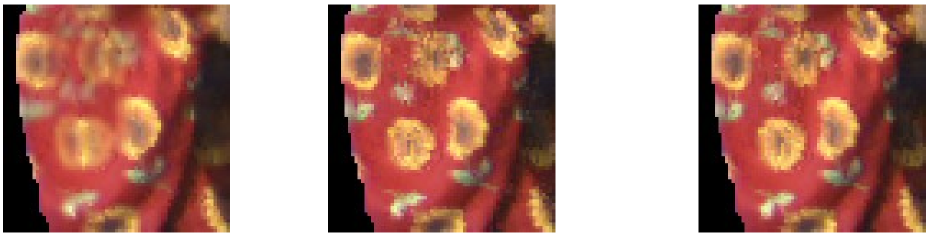


Figure 8: Comparison of rendering using the visual-hull(top), photo-hull(middle) and stereo(bottom)



(a) Artefacts in visual-hull (b) Refinement via photo-hull (c) Further improvement via stereo

Figure 9: Close-ups of the stages of refinement showing reduction in artefacts

[5] J. Carranza, C. Theobalt, M. Magnor, and H.-P. Seidel. Free-viewpoint video of human actors. *Proceedings ACM SIGGRAPH*, 22(3):569–577, 2003.

[6] Kong Man Cheung, Simon Baker, and Takeo Kanade. Visual hull alignment and refinement across time: A 3d reconstruction algorithm combining shape-from-silhouette with stereo. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2003.

[7] A. Criminisi, J. Shotton, A. Blake, and P. Torr. Gaze manipulation for one-to-one teleconferencing. In *ICCV*, pages 191—198, 2003.

[8] W. Bruce Culbertson, Thomas Malzbender, and Greg Slabaugh. Generalized voxel coloring. In *International Workshop on Vision Algorithms*, Corfu, Greece, 1999.

[9] K. Daniilidis, J. Mulligan, R. KcKendall, D. Schmid, G. Kamberova, and R. Bajcsy. Real-time 3-D Teleimmersion. *Kluwer*, pages 253—266, 2000.

[10] Jean-Sébastien Franco and Edmond Boyer. Exact polyhedral visual hulls. In *Fourteenth British Machine Vision Conference (BMVC)*, pages 329–338, September 2003. Norwich, UK.

[11] B. Goldluecke and M. Magnor. Space-Time Isosurface Evolution for Temporally Coherent 3D Reconstruction. In *CVPR*, pages S–E, Washington, D.C., USA, July 2004. IEEE Computer Society, IEEE Computer Society.

[12] O. Grau. A studio production system for dynamic 3d content. In *Proceedings of Visual Communications and Image Processing, Proceedings of SPIE*, 2003.

[13] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

[14] T. Kanade and P. Rander. Virtualized reality: Constructing virtual worlds from real scenes. *IEEE MultiMedia*, 4(2):34—47, 1997.

[15] Kiriakos N. Kutulakos and Steven M. Seitz. A theory of shape by space carving. Technical Report TR692, University of Rochester, 1998.

[16] A. Laurentini. The visual hull concept for silhouette-based image understanding. *IEEE Trans. Pattern Anal. Mach. Intell.*, 16(2):150–162, 1994.

[17] Wojciech Matusik, Chris Buehler, Ramesh Raskar, Steven J. Gortler, and Leonard McMillan. Image-based visual hulls. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 369–374. ACM Press/Addison-Wesley Publishing Co., 2000.

[18] S. Moezzi, L-C. Tai, and P. Gerard. Virtual view generation for 3d digital video. *IEEE MultiMedia*, 4(2):18—26, 1997.

[19] S. Seitz and C. Dyer. Photorealistic scene reconstruction by voxel coloring. In *Proc. CVPR*, pages 1067–1073, 1997.

[20] J. Starck and A. Hilton. Model-based multiple view reconstruction of people. In *IEEE International Conference on Computer Vision*, pages 915–922, 2003.

[21] J. Starck and A. Hilton. Virtual view synthesis from multiple view video. *Submitted Journal of Graphical Models*, 2003.

[22] R. Szeliski. Real-time octree generation from rotating objects. Technical report, Cambridge Research Laboratory, HP Labs, 1990.

[23] S. Vedula, S. Baker, and T. Kanade. Spatio-temporal view interpolation. *Eurographics Workshop on Rendering*, pages 1–11, 2002.

[24] C.L. Zitnick, S.B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski. High-quality video view interpolation using a layered representation. In *SIGGRAPH*, pages 600—608, 2004.