TOWARDS FREE-VIEWPOINT VIDEO CAPTURE IN CHALLENGING ENVIRONMENTS FOR COLLABORATIVE & IMMERSIVE ANALYSIS

Anton Frolov[1], Gareth Rendle[2], Adrian Kreskowski[2], Mariya Kaisheva[1], Bernd Froehlich[2], Volker Rodehorst[1]

[1] Computer Vision in Engineering, Bauhaus-Universität Weimar, Germany,
[2] Virtual Reality and Visualization Research Group, Bauhaus-Universität Weimar, Germany
{firstname}.{lastname}@uni-weimar.de

KEY WORDS: Free-Viewpoint Video, Outdoor Capture, Spatio-Temporal Reconstruction, 4D Model Rendering, Virtual Reality

ABSTRACT:

The ability to capture and explore complex real-world dynamic scenes is crucial for their detailed analysis. Tools which allow retrospective exploration of such scenes may support training of new employees or be used to evaluate industrial processes. In our work, we share insights and practical details for end-to-end acquisition of Free-Viewpoint Videos (FVV) in challenging environments and their potential for exploration in collaborative immersive virtual environments. Our lightweight capturing approach makes use of commodity DSLR cameras and focuses on improving both density and accuracy of Structure-from-Motion (SfM) reconstructions from small sets of images under difficult conditions. The integration of captured 3D models over time into a compact representation allows for efficient visualization of detailed FVVs in an immersive multi-user virtual reality system. We demonstrate our workflow on a representative acquisition of a suction excavation process and outline a use-case for exploration and interaction between collocated users and the FVV in a collaborative virtual environment.

1. INTRODUCTION

In recent years, advanced capturing and temporal integration techniques have been developed to produce high-quality geometry-based Free-Viewpoint Videos (FVV) (Lee et al., 2015), especially in the field of human performance capture (Prada et al., 2017) and usually under controlled lighting conditions (Collet et al., 2015, Guo et al., 2019). At the same time, the ability to capture FVVs outside of controlled conditions of studios and laboratories is highly desirable. The ability to capture in more challenging conditions would facilitate diverse applications of FVV, including collaborative and immersive analysis of complex real-world processes in multi-user virtual reality based on stereoscopic projection systems (Kulik et al., 2011).

Such systems enable domain experts to gather in a shared physical space, which is seamlessly extended by a virtual environment, into which additional visual content can be embedded. In addition, remote users can participate as avatars in a distributed representation of the scene (Kreskowski et al., 2020) to review, discuss, and explore complex recorded processes.

To enable practical acquisition of model-free FVVs for later analysis, we explore challenges of Structure-from-Motion (SfM) reconstruction of dynamic scenes from small sets of images captured in uncontrolled conditions with commodity DSLR cameras. We address the calibration approach in this particular context and argue for accurate pre-calibration of camera intrinsics.

We also consider issues often encountered in practice during image acquisition, such as optical defocus and sub-optimal illumination of surfaces. These are especially difficult to notice in outdoor campaigns, when human evaluation of every taken image is difficult or infeasible. To ensure that such adverse effects do not affect our imagery, we devise practical mechanisms for image quality control. We aim at automated identification of these conditions so they can be prevented during acquisition.

To enable re-exploration of captured FVVs, we design a practical approach for integration of the model into a compact representation suitable for out-of-core streaming. We demonstrate our approach on a use case of documenting an industrial process, which, once captured, can be explored by multiple users through virtual navigation and interaction metaphors. The chosen process involves operation of a suction excavator applied to realistic construction site materials (see Figure 1). Such documentation can be valuable for operators controlling the suction nozzle as evidence of non-destructive operation. Captured FVVs can also serve as training or evaluation materials.

With this paper we contribute insights and practical details aimed at capturing, encoding and visualizing industrial processes as FVV. Our acquisition methodology is adapted to reflect the specifics of spatio-temporal reconstruction from a small set of images under challenging conditions, while our approach to visualization allows immersive collaborative exploration of FVVs in a multi-user virtual reality system.

2. RELATED WORK

Here, we provide a brief overview of image-based methods for reconstruction of dynamic scenes over time and consider contributions focused on large 4D model visualization.

2.1 Image-based 3D reconstruction

Reconstruction of 3D models from 2D images can be regarded as one of the core photogrammetry (Albertz, 2009) and computer vision tasks dating back to the 1970s (Ullman, 1979), and is still a vital research area today. Below, we distinguish primarily between contributions on *online* and *offline* reconstruction methods.

Recently, real-time reconstruction of dynamic models with calibrated Multi-view Stereo (MVS) setups has received increasing attention (Dou et al., 2016, Dou et al., 2017). While online

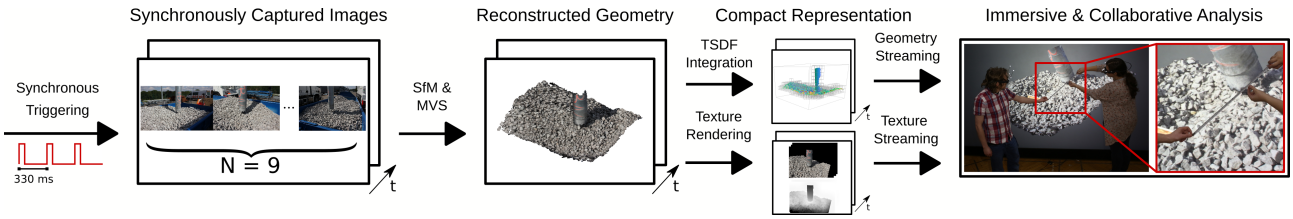


Figure 1. Overview of our Free-Viewpoint Video acquisition and analysis pipeline. Time-series of synchronous images are acquired using a lightweight capturing setup consisting of nine pre-calibrated DSLR cameras. After offline reconstruction, the individual 3D models are integrated into a compact Truncated Signed Distance Field representation. The compact models are streamed into our collaborative multi-user virtual environment for real-time geometry extraction. The extracted geometry is projectively textured on-the-fly to enable a convincing exploration of the acquired dynamic processes using virtual and mixed reality interaction and navigation techniques.

reconstruction pipelines have demonstrated impressive results, our use case focuses on retrospective analysis of dynamic processes, and as such affords additional processing time for reconstruction. We therefore focus on offline reconstruction methods for the remainder of this section.

Offline 3D reconstruction. Offline reconstruction methods are preferred when a high degree of geometric detail is required. In combination with SfM, MVS is a prevailing technique for dense 3D surface reconstruction (Furukawa and Ponce, 2010). A detailed overview of SfM methods can be found in (Schönberger and Frahm, 2016), and a general taxonomy of MVS methods has been proposed in (Seitz et al., 2006). Relying on bundle adjustment (Triggs et al., 1999), these techniques allow for simultaneous estimation of both the camera parameters and 3D geometry. Due to their inherent robustness, SfM methods are applied to large collections of possibly unordered imagery such as community photo collections (Agarwal et al., 2011). Practical convenience with respect to data acquisition makes SfM reconstruction methods a popular solution to 3D reconstruction, including reconstruction in challenging conditions.

Different implementations of SfM and MVS offer general and universal solutions to the task of 3D reconstruction, e.g. Agisoft Metashape (Agisoft LLC), RealityCapture (Epic Games) and Pix4Dmapper (Pix4D SA). At the same time, their application to specific scenarios and setups may offer space for improvement.

Offline 4D reconstruction for Free-Viewpoint Video. FVV allows users to observe reconstructions of dynamic scenes from arbitrary viewpoints. Image-based FVV methods synthesize novel viewpoints by interpolating between captured camera views (Germann et al., 2012). These methods are often limited to views located between camera poses, and require high density of coverage for good results.

Geometry-based FVV methods traditionally capture dynamic scenes with multiple RGB cameras, before extracting a 3D representation of the scene using MVS (Kanade et al., 1997) or by determining the volumetric occupancy (Moezzi et al., 1997) or visual hull (Matusik et al., 2000) of foreground objects. Current state-of-the-art FVV creation approaches capture sequences in controlled environments covered by 30+ cameras (Collet et al., 2015, Morgenstern et al., 2019). To form temporally coherent subsequences, models are non-rigidly deformed to fit adjacent time steps. The recent *Relightables* approach (Guo et al., 2019) obtains high-fidelity reflectance maps by capturing in a light stage environment using multiple novel high-resolution depth sensors, thereby increasing realism when reconstructed avatars are placed in arbitrarily illuminated virtual scenes. The above methods for FVV creation produce high-quality results, but require specialized camera configurations and lighting conditions, meaning that they do not transfer well to outdoor environments.

Efforts have been made to create FVV in more challenging outdoor environments such as sports stadia, where specialized methods deal with sparse, moving camera configurations by relying on pitch markings (Hilton et al., 2011) or image features (Germann et al., 2012) for reconstruction. While embedding of ground control points using visible markers is always possible, we focus our efforts on non-intrusive and lean capture, which ideally should not interfere with or modify the scene.

2.2 Efficient visualization of 4D models

Rendering high-resolution, time-varying geometry requires careful handling of a large volume of data, as the memory footprint of reconstructed sequences commonly exceeds the capacity of RAM and video RAM. As a result, 4D rendering systems often seek to compress data, or employ out-of-core approaches, which stream data from disk to the CPU and GPU when required.

Various 3D geometry representations have been augmented to allow efficient rendering of time-varying sequences. Compressed *point clouds* can represent each time step (Hosseini and Timmerer, 2018, Subramanyam et al., 2020), but are still memory intensive due to their explicit representation of each point's position. Temporally-coherent *triangle meshes* encode a dynamic sequence as meshes with consistent connectivity but varying vertex positions (Shinya, 2004). Changing mesh topology can be handled by periodically refreshing connectivity and texture data (Collet et al., 2015), or by updating mesh connectivity and texture regions incrementally when required (Prada et al., 2017).

Voxel representations such as Sparse Voxel Octrees (SVOs) (Laine and Karras, 2010) have been able to store detailed voxel occupancy information in a compact manner when treated as a Directed Acyclic Graph (DAG) by leveraging voxel patterns that appear in multiple spatial (Kämpe et al., 2013) and temporal (Kämpe et al., 2016) locations. While SVOs only encode binary occupancy of voxels, the Truncated Signed Distance Field (TSDF) representation (Curless and Levoy, 1996) can be used to implicitly encode surfaces in a set of voxels. Recent work has shown that bricks of TSDF voxels can be compressed effectively by converting them into a lower dimensional latent space, using Principal Component Analysis (PCA) (Canelhas

et al., 2017, Tang et al., 2018) or by training an encoder-decoder neural network (Tang et al., 2020).

3. ACQUISITION METHOD

Recent technological progress and cost reduction have made high-resolution imaging sensors, such as DSLR cameras, commercially available to private individuals and small enterprises. This has allowed reconstruction of highly detailed static geometries with just a single camera, with quality often comparable to or even exceeding other more expensive technologies such as 3D laser scanning.

Static reconstruction techniques such as SfM can also be applied to the reconstruction of time-varying dynamic scenes. This means that $N \geq 2$ separate cameras must be deployed and each one must synchronously capture a single image per reconstruction time step.

While both fast and accurate sensor configurations have been deployed before in controlled studio and lab conditions, outdoor scenes can present strong challenges to established sequential SfM workflows. In this section we address two potential issues, which we have found to appear in practice.

First, optimal coverage of the scene is difficult to implement, which means that frequent adaptation of camera settings and poses can be necessary. Such changes can result in sub-optimal imaging conditions. For example, images can be under- or over-exposed through shadows, reflections or direct light. During outdoor acquisitions, changing weather conditions can lead to rapid changes in the illumination conditions. Additionally, optical focus on surfaces may be lost, resulting in blurred images. Both of these effects will lead to reconstruction artifacts, such as holes in the model or distortions caused by a compromised sensor alignment.

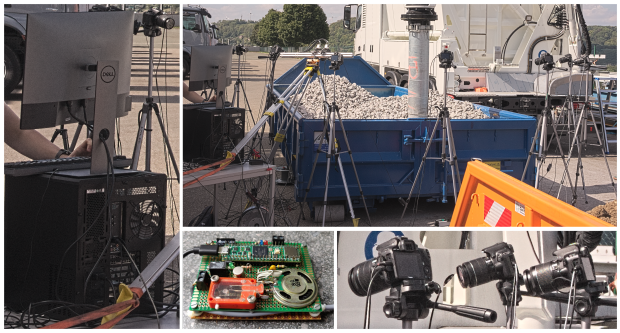
Secondly, many approaches assume that multiple images were captured by a single sensor exhibiting constant camera intrinsics, such as focal length, principal point and distortion model. Such approaches subsequently constrain the bundle adjustment such that camera intrinsics are estimated jointly. Severe failures in sensor alignment and estimation of 3D points are then inevitably present in reconstruction results. Even when the algorithm is allowed to solve for each camera model separately, a single registered image per sensor is rarely enough in practice to ensure an accurate fit.

In summary, sub-optimal imaging conditions and under-constrained variation of camera intrinsics between images of each single time-step will act as a bias on the quality of SfM reconstructions. Therefore, it makes sense to find a way to remove this bias and ensure that the problem is well-constrained, especially when the reconstruction takes place in adversarial conditions with low density of coverage.

We improve the results of standard SfM workflow by (a) reinforcing constraints on the sparse bundle adjustment problem by acquiring subpixel-accurate calibration of each individual sensor; and (b) asserting that images are both sharp and well-exposed through heuristic checks.

3.1 Continuous capture with commodity DSLR cameras

We demonstrate our approach on a test setup of nine Canon EOS 100D DSLR cameras configured to capture a construction site process. To ensure synchronous operation, we have supplied the DSLRs with a centralized hardware triggering and data transfer mechanism, which allowed us to simultaneously acquire nine perspectives of the scene at once (see Figure 2). In this way, we were able to achieve a stable imaging rate of 3 frames per second at 18 MP.



(a) (b) (c)

Figure 2. Deployment of nine DSLR cameras. Images from all sensors are stored distributed on SD-cards and consolidated at the acquisition control PC (a) via USB interface on-demand. The shutter of all sensors is engaged synchronously (tolerance $\Delta t \approx 100\mu s$) with a central triggering device (b) implemented using a Teensy controller. The trigger signal reaches each individual DSLR (c) through a wired hardware interface, which is relay-decoupled from other interfaces to avoid cross-talk.

While considerably higher imaging rates and spatial resolutions are attainable with modern-day scientific and studio production-grade sensors, commodity DSLRs are relatively easy to acquire and control without additional infrastructure. Since both the challenges we address and the approach we describe are equally relevant for both high-end and consumer-range sensors, we find it sufficient to evaluate our approach using this simple setup.

3.2 Camera pre-calibration

Calibration of camera parameters is an essential first step for a 3D reconstruction pipeline. In our case, accurate camera calibration allows us to constrain the bundle adjustment and avoid distortions in our resulting reconstructions.

Typically, calibration involves an estimation of a calibration matrix K and a set of distortion parameters $D = \{d_1, ...d_M\}$ on the basis of various types of calibration objects (Zhang, 2000, Bouguet, 2015). The matrix K is used to transform a point, expressed in the camera coordinate system, to a point in the image plane, while D is estimated based on the chosen distortion model such that the straight line preserving projection K remains unaffected by non-linear optical distortion.

In some applications, where high accuracy of camera calibration is not required at the edges and corners of the image, only a limited number of distortion parameters is estimated, thus simplifying the fitting task. It is very often the case that only the low-order, most influential radial distortion coefficients are estimated. Applications targeting higher accuracy fit more complex models of optical distortion, which normally require many high-quality images capturing the calibration object from varying perspectives.

Because of its practicality and reduced performance overhead we have chosen an 8-parameter distortion model (3 radial and 2 tangential distortion coefficients). We evaluate the parameters

	Number of triangulated points	RMS reprojection error, pix	Triangulated points, variance vector norm, m^2 x 10^{-4}		Camera centers, variance vector norm, m^2 x 10^{-4}	
			Mean	SD	Mean	SD
Uncalibrated	49614	**0.202**	37.035	8.113	39.710	24.751
Pre-calibrated	**53525**	0.253	**5.825**	**5.224**	**2.189**	**0.971**

Table 1. Statistical analysis of bundle adjustment results from a single reconstruction step. Uncertainty is given as variance vector magnitude in squared meters x 10^{-4}. Far points (> 10m) were removed from consideration. Pre-calibrated case demonstrates clear improvement through reduced overall uncertainties of both triangulated points and camera locations.

using the calibration pattern model proposed in (Schops et al., 2020), due to the versatility and robustness of its constituent element. To achieve a reliable fit with subpixel reprojection error we continuously capture a massive number of images by each individual camera from highly varying perspectives.

The impact of the accurate calibration is manifested highly in the corners of the image through improved density and spatio-temporal stability of estimated disparities in later stages (see section 4.1). Quantitative evaluation of reconstruction uncertainty (see Table 1) offers evidence on the advantages of the pre-calibrated approach.

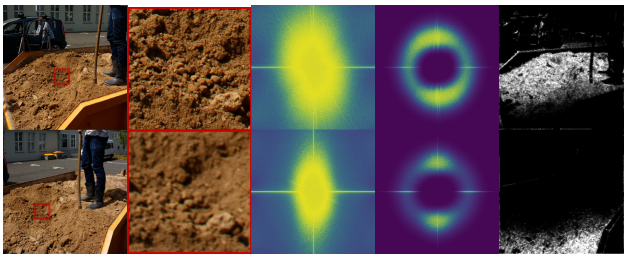
3.3 Image quality assessment

To attain the highest possible detail and accuracy of SfM reconstructions, it is critically important to ensure that each individual image is sharp and well-lit. While controlled environments such as motion capture studios offer stable imaging conditions, outdoor acquisitions often require continuous adaptation of exposure time, aperture and gain, while scene dynamics may require changes in camera placement and orientation. For such outdoor acquisition scenarios we have devised practical mechanisms for image quality control, which we have implemented in software and deployed during our acquisition campaigns.

Figure 3. A well-lit image (left, blue histogram) has a stronger presence in the preferred intensities range of the linear pixel intensities histogram (right, green) in comparison to the underexposed image (middle, orange histogram). This is reflected in a higher fitness score ν for a well-lit image.

First, we setup the equipment, focusing on target surfaces. We simultaneously take a single image with all deployed cameras and transfer all images on the acquisition control PC. Next, we make sure that all of the following conditions are satisfied:

Illumination conditions. To evaluate whether the image sensor was optimally illuminated, we capture a single RAW image with each camera and perform demosaicing. Further, we average the linear intensities from three color channels into a single grayscale channel and build a histogram of pixel intensities h_i (see Figure 3). Such histogram allows us to measure whether the captured intensities were in the preferred range $[I_{min}, I_{max}]$. The preferred range of intensities is chosen such that the pixel sensors operate linearly. In practice, I_{min} and I_{max} are chosen to be 20% and 80% of the pixel intensity range.



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

Figure 4. Defocus localization in sample images (a). The region of interest (b) in the source image appears sharp (top) and blurred (bottom). Notice that the Fourier amplitude spectrum (c) of a blurred image is more uniform, with its energy concentrated in the lower frequency range. The frequency range associated with sharp imagery (d) is filtered out with a smooth filter to avoid ringing artifacts. A SLIC-aggregated defocus response map (e) indicates the effective range of focus at each superpixel.

We then compute a fitness score ν, which denotes the share of pixels in the preferred range $[I_{min}, I_{max}]$. We eventually assert that ν is not less than a certain ν_{min}, which guarantees that at least a known portion of the image is captured under optimal illumination conditions.

Defocus localization. To ensure sharpness of captured imagery, we were able to empirically establish the following method for defocus localization. We perform a 2D Discrete Fourier Transform (DFT) on each individual image, which allows us to filter out the band of frequencies associated with highly-textured surfaces. When such surfaces are not in focus, this band of frequencies in amplitude spectrum is suppressed. In practice, for 18 MP images we have established such range to be $[769, 1024]$. A smooth 8-th order Butterworth filter is applied to avoid hard-filtering-related ringing artifacts. We then perform an Inverse DFT and extract the per-pixel defocus response map. To enhance human perception of the resulting defocus map, we perform Simple Linear Iterative Clustering (SLIC) (Achanta et al., 2012) and aggregate a response score for each superpixel by averaging. Using the defocus map it is possible to estimate whether the chosen focused distance is close to optimal (see Figure 4).

4. MODEL GENERATION

During acquisition, we capture a massive number of images, which, at this point, are corrected for distortion and annotated with a timestamp, sensor orientation and camera calibration matrix. Our subsequent goal is the generation of dense geometric models, as well as preparation of data structures and representations which allow for efficient access at rendering time.

4.1 Point-cloud and mesh reconstruction

For each separate timestamp of DSLR imagery we now triangulate a dense point cloud. For routine tasks of SfM and MVS

we use commercial software[1]. We force sensor poses and calibrations from (3.2) to be fixed during reconstruction, separately for each individual camera. Through the use of accurate calibration for each sensor we are able to visibly improve the results (see Figure 5).

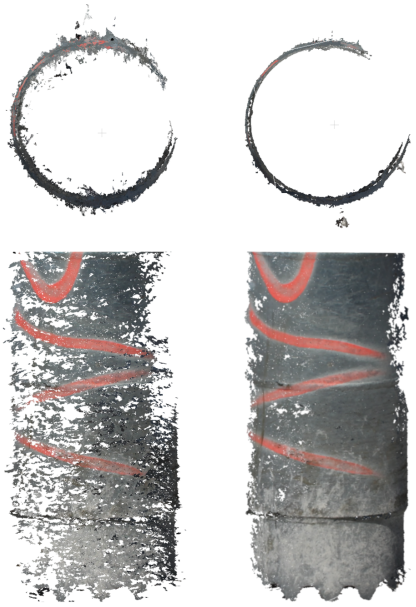


Figure 5. Unfiltered dense point cloud for calibration-free (left) and pre-calibrated case (right).

We produce textured triangle meshes in a dense multi-view geometry reconstruction step. While our setup allows for resolution of up to 90 million triangles, we choose to further optimize and decimate the initially over-tessellated meshes to a more practical resolution of 1 million triangles per 3D model.

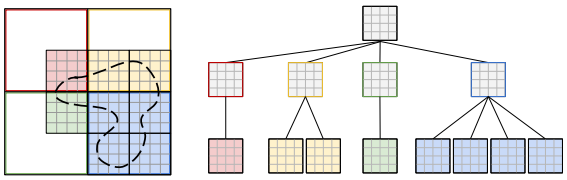
4.2 Free-Viewpoint Video encoding

To encode geometry in a compact form appropriate for streaming, the mesh representing each time frame of the spatio-temporal recording is transformed into a Sparse Brick Octree (SBO), where bricks consist of b^3 voxels containing TSDF values. This structure, shown in Figure 6, will be referred to as the TSDF-SBO. TSDF voxels implicitly encode geometry by storing a signed distance to the nearest surface, where the sign determines whether the voxel is inside or outside the surface. Signed distances with absolute value above threshold d_{max} are truncated, leaving a sparse set of voxels, and therefore a sparse set of bricks, which encode each time step. This sparse encoding means that we benefit from the compact nature of both sparse octree structures *and* the TSDF representation when storing and streaming spatio-temporal sequences, while the use of a spatial hierarchy enables rendering at varying Level-of-Detail (LOD).

To begin building the data structure, a bounding volume is computed that encompasses all meshes in the time series. The maximum LOD defines the depth of the TSDF-SBOs.

Each mesh is then converted to a TSDF-SBO. Triangles are processed in parallel to find voxels that lie within d_{max} of the mesh. Signed distances are calculated for each voxel, signed according to the normal vector of the nearest triangle. Voxels are then inserted into an octree. Octree nodes are added at all

[1] Agisoft Metashape Pro 1.6



(a) (b)

Figure 6. Simplified example of a TSDF surface encoding in 2D. A surface, dashed, is encoded in sparse bricks of TSDF voxels (a), which form a level-of-detail hierarchy (b).

tree levels where occupied voxels are present, and TSDF data bricks are associated with octree nodes. To encode octree structure, all nodes carry a child occupancy mask and a pointer to their first child. Children are stored sequentially. Since TSDF-SBOs from different frames are not interdependent, they can be processed independently in an out-of-core manner.

In addition to generating TSDF-SBOs for each time step, we calculate a global octree structure, which contains all nodes that are occupied throughout the entire sequence. This structure allows TSDF bricks to be assigned to a spatial location, via an index that corresponds to a node in the global octree structure.

To obtain color and depth images for the projective texturing stage described in Section 5.3, we render each reconstructed mesh from the perspective of the capturing cameras and store the images in a lightweight format. Color textures are compressed using the DXT1[2] standard. To avoid strong compression artifacts in the depth textures, we render and store lower resolution depth data without further compression.

5. IMMERSIVE EXPLORATION OF 4D PROCESSES

To support interactive frame rates, we stream texture and implicit geometry data to multiple rendering contexts, extracting geometry on the fly. Our system is implemented in C++, and uses the OpenGL Shading Language (GLSL) and graphics API.

5.1 Out-of-core streaming

Large amounts of data are needed to render an entire 4D sequence. In our implementation, each frame requires around 35MB of TSDF and image data, therefore requiring 6GB per minute of FVV. CPU memory constraints mean that loading longer sequences into RAM is often impractical. We therefore stream geometry and textures from SSD during playback. While out-of-core streaming incurs extra costs during run time when compared with 'up-front' loading of the sequence, these costs can be distributed sensibly to avoid negative impacts on the viewing experience. Furthermore, streaming out-of-core enables viewing of sequences of arbitrary lengths.

The applied out-of-core loading strategy assumes sequential playback of frames in either a forward or backward direction. A loader thread places all frames in a priority queue, ordered depending on the playback direction and the last frame requested by the rendering thread. Frames that are more likely to be requested next are assigned a higher priority. The loader thread ensures that TSDF-SBO structures for the first n frames in the queue are loaded into main memory when needed.

[2] khronos.org/opengl/wiki/S3_Texture_Compression

Rendering geometry from our TSDF-SBO format depends on a triangle extraction stage, described in Section 5.2. The extraction stage requires TSDF data to form the surface topology, as well as a spatial location linking each TSDF brick to a node in the global octree structure. TSDF bricks and location IDs are extracted from the TSDF-SBO that encodes each frame using Algorithm 1. This process is performed ahead-of-time on a thread parallel to the rendering thread.

Color and depth textures are also loaded asynchronously to reduce CPU to GPU transfer times.

Algorithm 1: Selection of TSDF bricks and brick location IDs from TSDF-SBO and global SVO structure

```
input : Desired level-of-detail l
input : TSDF-SBO structure T
input : Global SVO structure G
output: List of TSDF bricks B
output: List of brick location IDs L
// Recursive call traverses G and T in tandem
Function processNode(node, depth, global_ID):
    if depth < l then
        for i = 0 to 7 : do in parallel
            if node.hasChild(i) then
                child ← node.getChild(i);
                child_ID ← G.getChildID(global_ID,i);
                processNode(child, depth +1, child_ID);
            end
    else
        B.append(node.getTSDFBrick());
        L.append(global_ID)
    end
    return;
node ← T.getRootNode();
depth ← 0;
global_ID ← 0;
processNode(node, depth, global_ID);
```

5.2 Geometry extraction

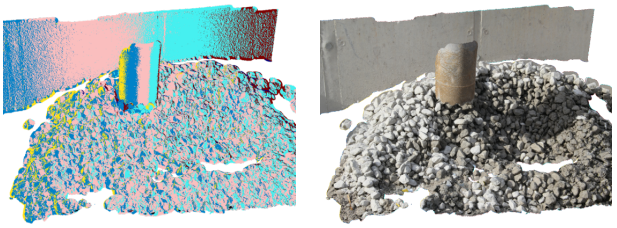
Triangles are extracted from TSDF data by finding the zero-value iso-surface using the trivially parallelizable Marching Cubes algorithm (Lorensen and Cline, 1987). Since TSDF data is uploaded to the GPU in non-overlapping bricks, vertex values of some cubes must be sampled from multiple TSDF bricks. Sampling the TSDF data at arbitrary points is possible by traversing the global octree structure that resides on the GPU.

Approximations of smooth per-vertex normals are derived during Marching Cubes extraction by averaging normal directions of adjoining triangles within a cube.

5.3 Visualization

Vertex positions and normals computed in the geometry extraction phase serve as input for a forward rendering pass. The geometry is textured in a projective manner (Debevec et al., 1998) based on intrinsic and extrinsic camera parameters estimated during the 3D reconstruction. Projective texturing, shown in Figure 7, avoids explicit association of textures with geometry, which would undermine the compact nature of TSDF-based representation. Color information is sampled per rasterized fragment from the color textures. For each fragment we sample the texture that corresponds to the camera aligning best with its surface normal. The DXT1-compressed color textures can be sampled directly in the fragment shading stage, since the decompression is performed in hardware.

To correctly resolve projective occlusions, pre-rendered depth maps are consulted at runtime. If the depth values of the best aligned camera indicate occlusion of current scene parts, we refer to the next best camera perspective.



(a) (b)

Figure 7. Projective texturing of a 3D model. The contribution of pre-rendered textures is based on the cosine similarity between surface normals and inverse projective camera directions. Different colors (a) indicate textures from different camera perspectives being sampled to determine the final color (b).

5.4 Experimental evaluation

To provide a concise summary of our system's performance, we capture a representative FVV with the aim of supporting teaching scenarios for prospective excavator operators. The FVV can also serve as process documentation, which can be immersively explored (see Figure 8) to investigate whether the excavation was executed according to the required standards.

Figure 8. Users working together in front of a multi-user virtual reality display. Real and virtual tools such as flashlights support the collaborative and immersive exploration of the FVVs.

Dataset. The FVV created to serve as a test dataset focuses on a suction excavator clearing a heap of gravel (see Figure 9). We pre-calibrate the cameras according to section 3.2 and reconstruct a 60 second FVV, consisting of 180 time steps. We render high-resolution color textures (2592×1728 pix per texture), as well as low-resolution depth textures (648×432 pix per texture) from the camera perspectives. For each time step, the geometry data, encoded as a TSDF-SBO, accounts for approximately 8.2 MB, while the color and depth textures account for 30.3 MB.

Rendering system. We render the FVV by integrating our visualization system into a multi-user virtual reality engine (Schneegans et al., 2014). When coupled with a multi-user projection system (Kulik et al., 2011), this allows up to six users to receive a unique stereoscopic perspective into the virtual world. Each user's perspective is rendered with a stereoscopic resolution of 4096×2160 pixels. Perspectives are rendered by a single *NVIDIA RTX Quadro 6000* GPU per user. The projection PC was equipped with an *Intel Xeon E5-2687W v4* running at 3.0 GHz.

Figure 9. Representative frames from our gravel excavation FVV. We refer the reader to our supplementary video material for a demonstration of interaction between users and the FVV in collaborative multi-user virtual reality.

	No Textures	1 Texture	5 Textures	9 Textures
T_R	1.6 ms	7.4 ms	10.5 ms	13.3 ms
T_U	4.1 ms	13.0 ms	23.6 ms	35.4 ms

Table 2. Average render frame (T_R) and upload frame (T_U) times for FVV rendered at stereoscopic 4096 x 2160 pix

Rendering performance and scalability. To evaluate the rendering performance, we profiled our application using a single GPU. We render a screen-filling view of our FVV, and measure the average frame time during the sequence for both upload and pure render frames (see Table 2). During upload frames, extra overhead is incurred from geometry extraction and texture copy operations. We estimate the cost of projective texturing by omitting a varying number of original camera representations and recording changes in the average frame time.

We observe that frame times increase sub-linearly when projective texturing is active. Nevertheless, it is apparent that when nine camera perspectives are used for texturing, our implementation is limited to a frame rate of around 28 Hz. To enable projective texturing from significantly larger numbers of cameras without losing interactivity at high display resolutions, video encoding could be explored for efficient streaming of color maps. It would also be possible to identify changing texture regions during pre-processing, to allow partial texture update at each frame. Alternatively, an implicit coupling of extracted geometry to a compact texture atlas could reduce texture upload overhead, as demonstrated in real-time model compression schemes (Tang et al., 2020). However, the challenge of creating texture maps that can be consistently applied across geometric levels-of-detail remains.

6. CONCLUSIONS & FUTURE WORK

In this work, we have presented an end-to-end system for capture and immersive collaborative analysis of FVV. We have described our practical acquisition and 3D reconstruction pipeline, which allows to capture dynamic scenes with high spatial detail even in challenging outdoor conditions. To assess the suitability of our FVV representation for interactive use, such as training and process documentation, we have evaluated both performance and scalability of our real-time immersive virtual reality environment. In our supplementary video[3] we provide an example of a verification scenario for correct operation of a construction site process. This verification is achieved interactively through collaborative analysis of captured FVVs.

Although interactive rendering frame-rates were achieved for high-resolution displays in our multi-user system, further output-sensitive rendering techniques could be developed to enable real-time texture handling for many more capturing perspectives.

[3] `https://youtu.be/9XVTGtlwBi4`

Temporal coherence can be exploited to improve both the initial reconstructions and the efficiency of FVV encoding (Kämpe et al., 2016, Prada et al., 2017). Through spatio-temporal modeling of motion and deformation of general objects, observations can be propagated between the temporal steps of the acquisition, resulting in more complete and coherent models. Combining inter-frame encoding techniques based on temporal coherence with our system would lead to the ability to represent larger, temporally denser dynamic real-world scenes.

ACKNOWLEDGEMENTS

This work has received funding from the German Federal Ministry of Education and Research (BMBF) under grant 02K18K020. In addition to our project partners, we thank Sven Daubert and Michael Spreer as well as the members of VRSYS at Bauhaus-Universität Weimar for their support.

REFERENCES

Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Susstrunk, S., 2012. SLIC Superpixels Compared to State-of-the-Art Superpixel Methods. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(11), 2274–2282.

Agarwal, S., Furukawa, Y., Snavely, N., Simon, I., Curless, B., Seitz, S. M., Szeliski, R., 2011. Building Rome in a Day. *Commun. ACM*, 54(10), 105–112.

Albertz, J., 2009. 100 Years German Society for Photogrammetry, Remote Sensing, and Geoinformation. *Photogrammetrie - Fernerkundung - Geoinformation*, 2009(6), 485-486.

Bouguet, J. Y., 2015. Camera calibration tool box for matlab. `http://www.vision.caltech.edu/bouguetj/calib_doc/`.

Canelhas, D. R., Schaffernicht, E., Stoyanov, T., Lilienthal, A. J., Davison, A. J., 2017. Compressed Voxel-Based Mapping Using Unsupervised Learning. *Robotics*, 6(3).

Collet, A., Chuang, M., Sweeney, P., Gillett, D., Evseev, D., Calabrese, D., Hoppe, H., Kirk, A., Sullivan, S., 2015. High-Quality Streamable Free-Viewpoint Video. *ACM Trans. Graph.*, 34(4).

Curless, B., Levoy, M., 1996. A volumetric method for building complex models from range images. *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, Association for Computing Machinery, New York, NY, USA, 303–312.

Debevec, P., Yu, Y., Borshukov, G., 1998. Efficient view-dependent image-based rendering with projective texture-mapping. G. Drettakis, N. Max (eds), *Rendering Techniques '98*, Springer Vienna, Vienna, 105–116.

Dou, M., Davidson, P., Fanello, S. R., Khamis, S., Kowdle, A., Rhemann, C., Tankovich, V., Izadi, S., 2017. Motion2fusion: Real-Time Volumetric Performance Capture. *ACM Trans. Graph.*, 36(6).

Dou, M., Khamis, S., Degtyarev, Y., Davidson, P., Fanello, S. R., Kowdle, A., Escolano, S. O., Rhemann, C., Kim, D., Taylor, J., Kohli, P., Tankovich, V., Izadi, S., 2016. Fusion4D: Real-Time Performance Capture of Challenging Scenes. *ACM Trans. Graph.*, 35(4).

Furukawa, Y., Ponce, J., 2010. Accurate, Dense, and Robust Multiview Stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(8), 1362-1376.

Germann, M., Popa, T., Keiser, R., Ziegler, R., Gross, M., 2012. Novel-View Synthesis of Outdoor Sport Events Using an Adaptive View-Dependent Geometry. *Computer Graphics Forum*, 31(2pt1), 325-333.

Guo, K., Lincoln, P., Davidson, P., Busch, J., Yu, X., Whalen, M., Harvey, G., Orts-Escolano, S., Pandey, R., Dourgarian, J., Tang, D., Tkach, A., Kowdle, A., Cooper, E., Dou, M., Fanello, S., Fyffe, G., Rhemann, C., Taylor, J., Debevec, P., Izadi, S., 2019. The Relightables: Volumetric Performance Capture of Humans with Realistic Relighting. *ACM Trans. Graph.*, 38(6).

Hilton, A., Guillemaut, J., Kilner, J., Grau, O., Thomas, G., 2011. 3D-TV Production From Conventional Cameras for Sports Broadcast. *IEEE Transactions on Broadcasting*, 57(2), 462-476.

Hosseini, M., Timmerer, C., 2018. Dynamic adaptive point cloud streaming. *Proceedings of the 23rd Packet Video Workshop*, PV '18, Association for Computing Machinery, New York, NY, USA, 25–30.

Kämpe, V., Rasmuson, S., Billeter, M., Sintorn, E., Assarsson, U., 2016. Exploiting coherence in time-varying voxel data. *Proceedings of the 20th ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, I3D '16, Association for Computing Machinery, New York, NY, USA, 15–21.

Kämpe, V., Sintorn, E., Assarsson, U., 2013. High Resolution Sparse Voxel DAGs. *ACM Trans. Graph.*, 32(4).

Kanade, T., Rander, P., Narayanan, P. J., 1997. Virtualized reality: constructing virtual worlds from real scenes. *IEEE MultiMedia*, 4(1), 34-47.

Kreskowski, A., Beck, S., Froehlich, B., 2020. Output-Sensitive Avatar Representations for Immersive Telepresence. *IEEE Transactions on Visualization and Computer Graphics*, 1-1.

Kulik, A., Kunert, A., Beck, S., Reichel, R., Blach, R., Zink, A., Froehlich, B., 2011. C1x6: A Stereoscopic Six-User Display for Co-Located Collaboration in Shared Virtual Environments. *ACM Trans. Graph.*, 30(6), 1–12.

Laine, S., Karras, T., 2010. Efficient sparse voxel octrees. *Proceedings of the 2010 ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, I3D '10, Association for Computing Machinery, New York, NY, USA, 55–63.

Lee, C.-C., Tabatabai, A., Tashiro, K., 2015. Free viewpoint video (FVV) survey and future research direction. *APSIPA Transactions on Signal and Information Processing*, 4, e15.

Lorensen, W. E., Cline, H. E., 1987. Marching cubes: A high resolution 3d surface construction algorithm. *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '87, Association for Computing Machinery, New York, NY, USA, 163–169.

Matusik, W., Buehler, C., Raskar, R., Gortler, S. J., McMillan, L., 2000. Image-based visual hulls. *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, ACM Press/Addison-Wesley Publishing Co., USA, 369–374.

Moezzi, S., Tai, L.-C., Gerard, P., 1997. Virtual View Generation for 3D Digital Video. *IEEE MultiMedia*, 4(1), 18–26.

Morgenstern, W., Hilsmann, A., Eisert, P., 2019. Progressive non-rigid registration of temporal mesh sequences. *European Conference on Visual Media Production*, CVMP '19, Association for Computing Machinery, New York, NY, USA.

Prada, F., Kazhdan, M., Chuang, M., Collet, A., Hoppe, H., 2017. Spatiotemporal Atlas Parameterization for Evolving Meshes. *ACM Trans. Graph.*, 36(4).

Schneegans, S., Lauer, F., Bernstein, A., Schollmeyer, A., Froehlich, B., 2014. guacamole - an extensible scene graph and rendering framework based on deferred shading. *2014 IEEE 7th Workshop on Software Engineering and Architectures for Realtime Interactive Systems (SEARIS)*, 35–42.

Schops, T., Larsson, V., Pollefeys, M., Sattler, T., 2020. Why having 10,000 parameters in your camera model is better than twelve. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2532–2541.

Schönberger, J. L., Frahm, J., 2016. Structure-from-motion revisited. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 4104–4113.

Seitz, S., Curless, B., Diebel, J., Scharstein, D., Szeliski, R., 2006. A comparison and evaluation of multi-view stereo reconstruction algorithms. *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, 1, 519–528.

Shinya, M., 2004. Unifying measured point sequences of deforming objects. *Proceedings. 2nd International Symposium on 3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004.*, 904–911.

Subramanyam, S., Viola, I., Hanjalic, A., Cesar, P., 2020. User centered adaptive streaming of dynamic point clouds with low complexity tiling. *Proceedings of the 28th ACM International Conference on Multimedia*, MM '20, Association for Computing Machinery, New York, NY, USA, 3669–3677.

Tang, D., Dou, M., Lincoln, P., Davidson, P., Guo, K., Taylor, J., Fanello, S., Keskin, C., Kowdle, A., Bouaziz, S., Izadi, S., Tagliasacchi, A., 2018. Real-Time Compression and Streaming of 4D Performances. *ACM Trans. Graph.*, 37(6).

Tang, D., Singh, S., Chou, P. A., Häne, C., Dou, M., Fanello, S., Taylor, J., Davidson, P., Guleryuz, O. G., Zhang, Y., Izadi, S., Tagliasacchi, A., Bouaziz, S., Keskin, C., 2020. Deep implicit volume compression. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 1290–1300.

Triggs, B., McLauchlan, P. F., Hartley, R. I., Fitzgibbon, A. W., 1999. Bundle adjustment - a modern synthesis. *Proceedings of the International Workshop on Vision Algorithms: Theory and Practice*, ICCV '99, Springer-Verlag, Berlin, Heidelberg, 298–372.

Ullman, S., 1979. The interpretation of structure from motion. *Proceedings of the Royal Society of London. Series B. Biological Sciences*, 203(1153), 405-426.

Zhang, Z., 2000. A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11), 1330-1334.