VRVideos: A flexible pipeline for Virtual Reality Video Creation

1st Anthony Dickson
Depart. of Computer Science
University of Otago
Dunedin, New Zealand
dican732@student.otago.ac.nz

2nd Jeremy Shanks
Depart. of Computer Science
University of Otago
Dunedin, New Zealand
shaje415@student.otago.ac.nz

3rd Jonathan Ventura
Depart. of Computer Science and Software Engineering
California Polytechnic State University
San Luis Obispo, USA
jventu09@calpoly.edu

4th Alistair Knott
School of Engineering and Computer Science
Victoria University of Wellington
Wellington, New Zealand
ali.knott@vuw.ac.nz

5th Stefanie Zollmann
Depart. of Computer Science
University of Otago
Dunedin, New Zealand
stefanie.zollmann@otago.ac.nz

***Abstract*—Recent advances in Neural Radiance Field (NeRF)-based methods have enabled high-fidelity novel view synthesis for video with dynamic elements. However, these methods often require expensive hardware, take days to process a second-long video and do not scale well to longer videos. We create an end-to-end pipeline for creating dynamic 3D video from a monocular video that can be run on consumer hardware in minutes per second of footage, not days. Our pipeline handles the estimation of the camera parameters, depth maps, 3D reconstruction of dynamic foreground and static background elements, and the rendering of the 3D video on a computer or VR headset. We use a state-of-the-art visual transformer model to estimate depth maps which we use to scale COLMAP poses and enable RGB-D fusion with estimated depth data. In our preliminary experiments, we rendered the output in a VR headset and visually compared the method against ground-truth datasets and state-of-the-art NeRF-based methods.**

***Index Terms*—Virtual Reality, Image Processing, Computer Vision, Reconstruction, Artificial Intelligence, Vision and Scene Understanding**

I. Introduction

Virtual Reality (VR) headsets and VR experiences gained a lot of attention in recent years. This development was in particular driven by the increasing affordability and accessibility of VR headsets in the consumer market. While they are still mostly used for gaming and content consumption, recently there is more interest in creating content for these devices—a process that typically requires special hardware or expertise. There are several devices for creating content for VR consumption such as stereo cameras. However, often they are expensive such as 360 6-DoF cameras or only allow for capturing 360 footage or 360+depth footage which does not provide a fully immersive experience. In addition, a lot of existing image content that has been captured already is difficult to consume on such devices.

In contrast, image synthesis and image-based rendering often work with monocular video input. In particular, light fields [5], layered representations [2] and Neural Radiance Field (NeRF) [6] have become popular in recent years. They come with the advantage of synthesising views that have not been seen in the original capture. However, they also come with a high computational cost. This is a particular problem for dynamic video data.

In our work, we work towards addressing the challenge of high computational cost by combining traditional geometric methods with machine learning-based depth estimation and a layered representation to capture dynamic contents. We propose an end-to-end pipeline for creating 3D content from monocular video that handles the estimation of the camera parameters, depth maps, 3D reconstruction of dynamic foreground and static background elements. In addition, our pipeline can render 3D video on a computer, mobile device or VR headset. Our pipeline is flexible as it can be run with monocular RGB video, RGB-D video, or RGB-D datasets with ground-truth pose data. This flexibility means that we can easily use the pipeline to compare the quality of the results in future experiments and are able to provide a flexible benchmarking system for the creation of VR videos.

In our preliminary experiments, we demonstrate how to use the output in a VR headset and inspect how the method visually compares against state-of-the-art NeRF-based methods.

II. Related Work

Our work is closely related to methods that aim to create 3D photographs and 3D videos. Furthermore, it relies on COLMAP, learned models for depth estimation and Truncated Signed Distance Function (TSDF) fusion.

A. 3D Photographs

In recent years there have been many methods proposed to create 3D photographs that provide immersive experiences from single images. One of the main characteristics of these 3D photographs is that they enable novel view synthesis with motion parallax. Hedman et al. [1] construct 3D panoramas from RGB-D data captured with a dual-camera iPhone that

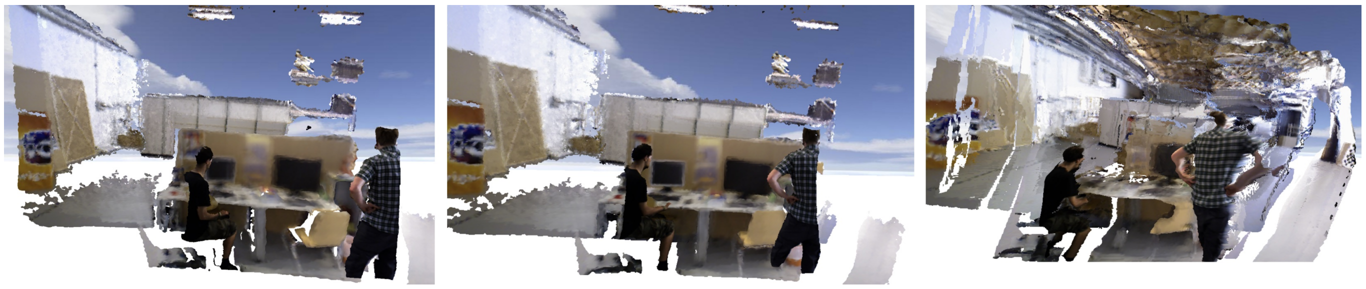

Fig. 1. VRVideos: Left) VR video created from ground truth (Actual depth data and pose data), Middle) VR video from actual depth data and computed pose data, Right) VR video from monocular video using estimated depth and estimated pose data.

exhibits realistic motion parallax. However, their method is only demonstrated on static scenes without any people. Niklaus et al. [9] and Kopf et al. [2] leverage learned depth estimation models and inpainting to produce realistic view synthesis. Niklaus et al. [9] focus on creating a 3D zoom effect and introduce a pipeline for improving off-the-shelf depth estimation models, addressing common issues with depth estimation models such as wavy walls.

B. 3D Video

While most of the above methods work on single images, Wang et al. [13] propose a method for creating 3D video from a pair of near-duplicate photos. Their method includes a neural network trained to synthesise novel views from layered depth images and 2D image features. This approach enables a plausible estimation of how appearance may change between the pair of photos. NeRF methods have been developed that can produce high-quality novel view synthesis from a set of a few photographs [6]. Recent methods are able to create NeRF representations within a few seconds [7]. NeRF-based methods have also been applied to video with dynamic elements [3], [4], [15]. The implicit representation inherent to neural networks helps address many of the challenges in reconstructing dynamic 3D scenes with estimated depth and TSDF fusion-based approaches. The main drawback of these methods is their expensive hardware requirements, long compute time and poor scaling with video length. In the work of Li et al. [3] training took one week on 8 NVIDIA Volta GPUs for a 10-second, 30 FPS clip.

Earlier methods have also been proposed to extract 3D representations from videos. Still, they are either trained for a specific use case such as soccer [11] or focus on static or rotating cameras [17].

In our work, we combine traditional geometric methods with machine learning-based depth estimation and a layered representation to capture dynamic contents to reduce the computation time and the complexity of the VR videos to make them suitable for replay in a VR headset.

III. 3D Video Pipeline

Our pipeline for creating and rendering VR video consists of four main steps: data preparation; creating a 3D representation of the static background; creating a 3D triangle mesh representation of the dynamic foreground for each frame of the video sequence; and rendering the 3D video (Figure 2). Our pipeline is created with the main purpose of taking monocular video as input but is also flexible enough to support TUM RGB-D datasets [12] or RGB-D datasets captured on an iPhone[1]. These two RGB-D formats are intended for comparing the quality of the 3D video when using different combinations of ground-truth and estimated data for depth and camera pose in our experiments.

A. Data Preparation

Our pipeline requires depth data, segmentation masks of dynamic foreground objects, and pose data for further processing. When the pipeline is only given RGB video, we need to compute these before further processing.

For depth estimation, we use the DPT depth estimation model [10] on each of the RGB frames. We use the model weights that were fine-tuned on the NYU dataset [8] and produce depth that roughly corresponds to meters (the model outputs depth in the interval [0, 10]).

For the segmentation of dynamic foreground, we create Detectron2 [14] instance segmentation masks for each frame of the video and derive binary masks for any people in the video.

In addition, we use COLMAP's automatic reconstructor pipeline to estimate the camera parameters. Since running COLMAP on many frames (e.g., several hundred) can take more than several hours, we use a 'frame step' parameter to uniformly sample a subset of frames to lower the runtime and interpolate the poses of the in-between frames. We use the instance segmentation masks to make sure that COLMAP ignores detected image features in dynamic areas.

B. Static Background Mesh Reconstruction

For computing the static background of the video, we use TSDF fusion [16]. TSDF is a 3D voxel volume where each voxel is labelled with its distance to the nearest surface that used for dense mesh reconstruction. We use this in combination with the estimated depth maps and instance segmentation to create a single static mesh for the background.

[1]https://www.strayrobots.io/components/stray-scanner-collect-rgb-d-datasets-on-iphone

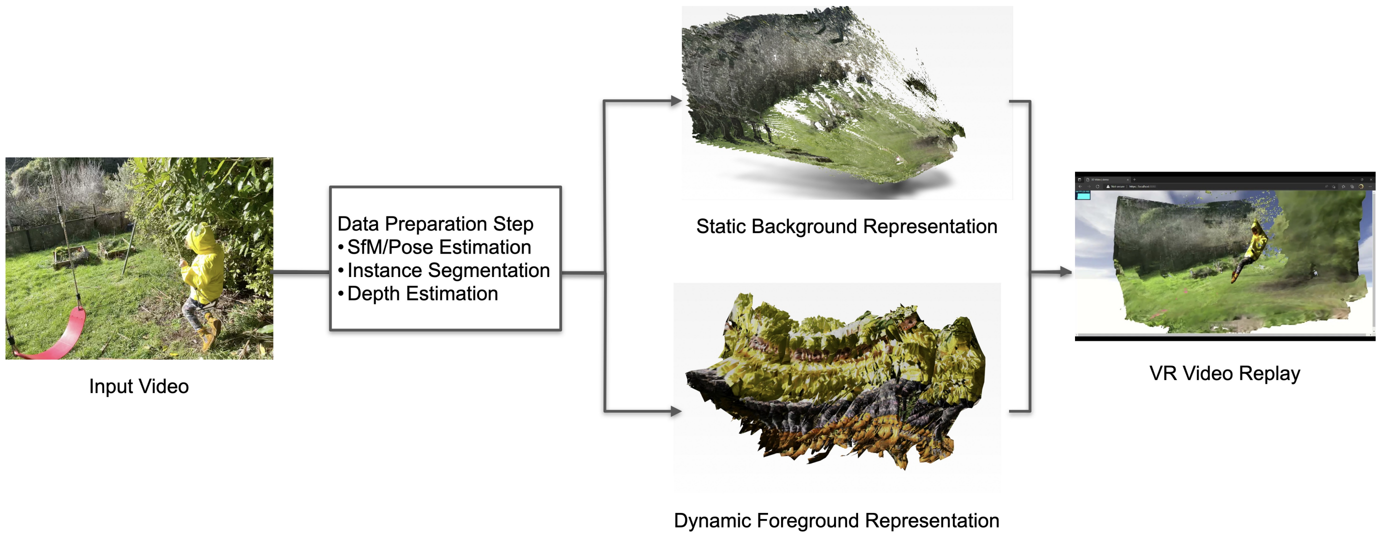

Fig. 2. Overview of our pipeline for creating a VR video. Our pipeline consists of four main steps: preparing the data; creating a 3D representation of the dynamic foreground for each frame of the video sequence; creating a 3D representation of the static background; and rendering the 3D video.

C. Dynamic Mesh Reconstruction

The next step is to compute a mesh representation of parts of the scene that change per frame. Our approach is based on the meshing technique from the paper "Soccer on Your Tabletop" [11] and creates a textured triangle mesh from an RGB frame, depth map, camera matrix, camera pose and instance segmentation mask.

We thereby create a 3D point cloud by projecting the 2.5D points (2D image coordinates + depth) into 3D world coordinates by applying the depth, inverse camera matrix and inverse pose to each 2D point in the image:

$$X_p = [R|t]^{-1}(d_p K^{-1} p)$$

where: p is the 2D pixel coordinates of a point in the current frame; X_p is p projected into 3D world coordinates; $[R|t]$ is the camera pose matrix for the current frame which is a concatenation of the rotation matrix R and the translation column vector t; d_p is the depth at p; K is the camera intrinsic parameter matrix.

The mesh faces are created by running Delaunay triangulation on the 2D points within the dynamic regions of a given frame. The faces are then filtered out if the corresponding 2D point of a vertex is more than 2 pixels away from the other vertices in the face, or if the corresponding depth value for a vertex is more than 10cm away from other vertices in the face. This filtering step removes stretched out mesh faces and parts of the background that were mislabelled as foreground.

Once the mesh geometry has been created, the next step is to map the texture and UV coordinates. We crop the RGB frame to the extent of the 2D points (minimum and maximum coordinates along each axis) and use this as the texture. The UV Coordinates are then the 2D points shifted so that they are relative to the top left corner of the texture.

Once all objects in the frame have been processed, the mesh data is merged into a single mesh object for the current frame.

D. Export and Rendering

To the best of our knowledge, there is no standard file format for mesh videos. We export the 3D video to a folder containing three files for the: foreground meshes, background mesh, and metadata file. We leverage the glTF file format to store the foreground meshes of all frames in a single file. Each mesh is labelled with its frame index since some video frames will not have any dynamic elements in them and thus have no mesh. This ensures that frames can be displayed at the correct time.

We implement a web-based application using THREE.js[2] for rendering the 3D videos to either a typical computer display or a VR headset. Our renderer is portable, performant and supports basic replay capabilities such as starting/stopping/forwarding the video. It runs on desktop computers, laptops, mobile devices and VR headsets at a steady 60 FPS.

IV. Preliminary Results

We used our web-based VR video renderer to demonstrate how the output of our pipeline can be replayed in a VR headset (Oculus Quest 2). We used this for first feasibility testing and quality inspection. Our VR video player allowed us to render with 60 FPS on the device which is suitable for immersive experiences. We also used the VR video player to inspect the quality that can be achieved from RGB input compared to the quality that can be achieved from RGB-D videos or RGB-D datasets with ground truth data (Figure 1) by using the TUM RGB-D dataset [12] that comes with RGB-D data and ground truth pose data. While the results showed that there is a reduction in visual quality, we are interested to analyse how much this decrease in quality affects the user experience.

In addition, we compared how the results of our method visually compare against state-of-the-art NeRF-based methods that take several days to compute (Figure 3). While we do not

[2]https://threejs.org

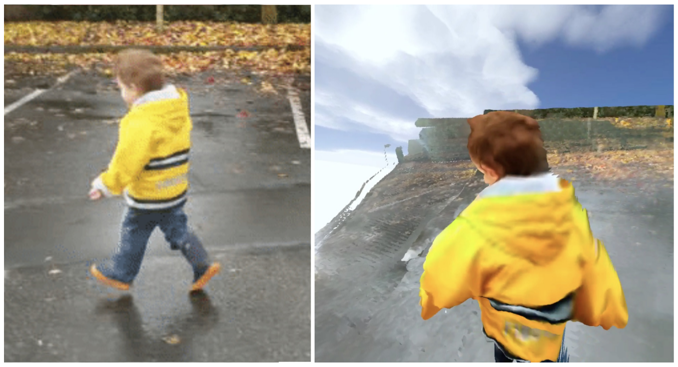
Fig. 3. Comparison of output between NSFF [4] and our pipeline on the 'kid running' sequence. Please note that we are not using image inpainting to fill in missing parts of the scene.

achieve the same visual quality and have more visual artefacts present, there is still a major advantage in that the results are created within a few minutes and rendered at interactive frame rates on a VR headset.

We make video comparisons available online for the pipeline using a sequence from the TUM RGD-D dataset[3] and our own dataset with ground-truth depth and pose data captured with an iPhone[4].

V. Conclusion and Future Work

In this work, we proposed an end-to-end pipeline for creating 3D content from a monocular video that handles the estimation of the camera parameters, depth maps, 3D reconstruction of dynamic foreground and static background elements. We demonstrated how our pipeline renders the created VR video on a computer or VR headsets with interactive framerates.

The next steps of our work include experiments that investigate the accuracy of single parts of this pipeline, such as pose data, estimated depth, and an in-depth investigation of how the output from our method compares against state-of-the-art NeRF-based methods and how users perceive the achieved quality.

Acknowledgment

We gratefully acknowledge the support of the New Zealand Marsden Council through Grant UOO1724 and the U.S. National Science Foundation through award #2144822.

References

[1] P. Hedman and J. Kopf. Instant 3d photography. *ACM Transactions on Graphics (TOG)*, 37(4):1–12, 2018.

[2] J. Kopf, K. Matzen, S. Alsisan, O. Quigley, F. Ge, Y. Chong, J. Patterson, J.-M. Frahm, S. Wu, M. Yu, et al. One shot 3d photography. *ACM Transactions on Graphics (TOG)*, 39(4):76–1, 2020.

[3] T. Li, M. Slavcheva, M. Zollhoefer, S. Green, C. Lassner, C. Kim, T. Schmidt, S. Lovegrove, M. Goesele, and Z. Lv. Neural 3d video synthesis. *arXiv preprint arXiv:2103.02597*, 2021.

[4] Z. Li, S. Niklaus, N. Snavely, and O. Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6498–6508, 2021.

[5] B. Mildenhall, P. P. Srinivasan, R. Ortiz-Cayon, N. K. Kalantari, R. Ramamoorthi, R. Ng, and A. Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. *ACM Trans. Graph.*, 38(4), jul 2019.

[6] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

[7] T. Müller, A. Evans, C. Schied, and A. Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, July 2022.

[8] P. K. Nathan Silberman, Derek Hoiem and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[9] S. Niklaus, L. Mai, J. Yang, and F. Liu. 3d ken burns effect from a single image. *ACM Transactions on Graphics (ToG)*, 38(6):1–15, 2019.

[10] R. Ranftl, A. Bochkovskiy, and V. Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12179–12188, 2021.

[11] K. Rematas, I. Kemelmacher-Shlizerman, B. Curless, and S. Seitz. Soccer on your tabletop. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4738–4747, 2018.

[12] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012.

[13] Q. Wang, Z. Li, D. Salesin, N. Snavely, B. Curless, and J. Kontkanen. 3d moments from near-duplicate photos, 2022.

[14] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019.

[15] W. Xian, J.-B. Huang, J. Kopf, and C. Kim. Space-time neural irradiance fields for free-viewpoint video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9421–9431, 2021.

[16] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1802–1811, 2017.

[17] S. Zollmann, A. Dickson, and J. Ventura. Casualvrvideos: Vr videos from casual stationary videos. In *26th ACM Symposium on Virtual Reality Software and Technology*, pages 1–3, 2020.

[3] https://youtu.be/Sd5yuztCyVE

[4] https://youtu.be/maxED7CCZNY