Practical 3D Photography

Johannes Kopf   Suhib Alsisan   Francis Ge   Yangming Chong   Kevin Matzen
Ocean Quigley   Josh Patterson   Jossie Tirado   Shu Wu   Michael F. Cohen
Facebook

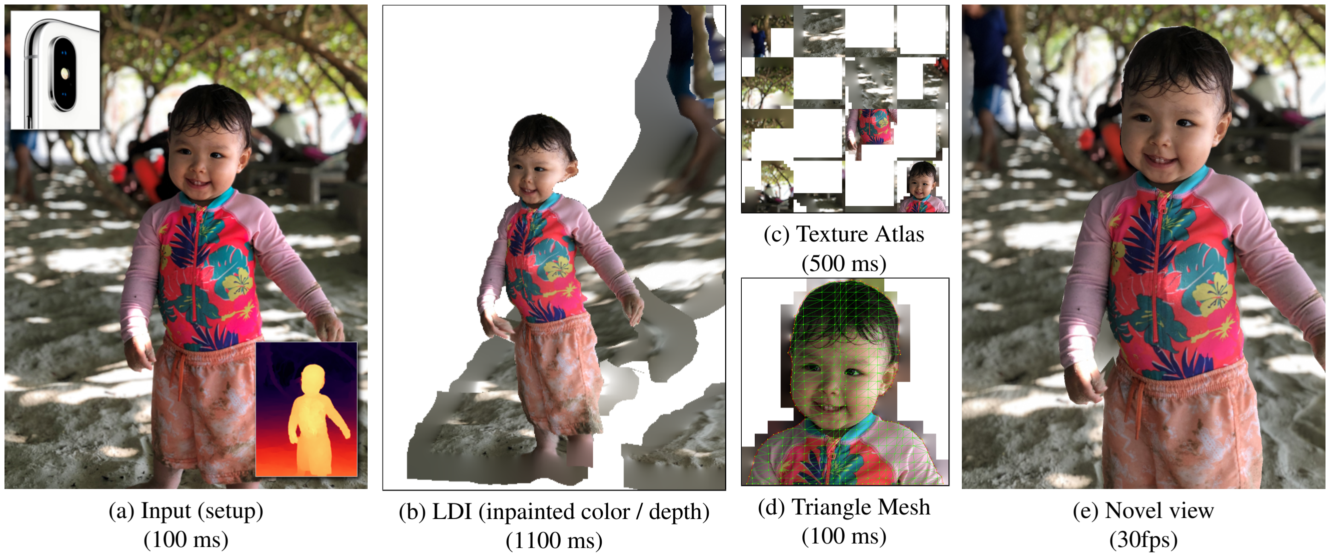

Figure 1. 3D Photo Creation. Runtime measured on iPhone X.

Abstract

We present the algorithmic choices that went into the creation and presentation of 3D Photos, a new medium viewed by millions of people. 3D Photos leverage a variety of modern computer vision and computer graphics technologies to efficiently process image+depth pairs to produce a compact and easy to share result. They exhibit parallax through virtual camera motion that is mapped to scrolling and/or device motion. In VR 3D Photos are rendered in stereo and respond to head motion. They are currently the only easily captured user-generated-content in VR.

1. Introduction

Traditional 2D photography lets us capture, with a single click, the world around us as an instant frozen in time. *3D photography* is a new way to make these captured moments come back alive: Although still static in time, 3D photos can be interactively explored by changing the viewpoint after the fact to more fully relive the moment. The ability to change viewpoint is possible on mobile devices, but enables even more life-like experiences in Virtual Reality, by facilitating stereo viewing with head-motion induced parallax.

However, capturing and displaying 3D photos poses challenges that are not present in 2D photography: in addition to color, depth per pixel is required, viewpoint changes reveal previously *occluded* parts of the scene which must be filled, and the affordances for changing viewpoint must be developed. Depth is now possible to acquire from multi-camera devices, however, accurately filling hidden content requires multiple captured views [3, 13]. This is difficult from a technical, and most importantly, from a practical standpoint: capturing multiple angles goes beyond the effort that most people are willing to spend on a photograph. Viewing 3D photos also requires an intuitive connection between the viewer and the media.

We present a practical approach to 3D photography, specifically we address five design objectives:

Effort: the capture should occur in a *single shot* and not require any special hardware. This effectively restricts us to capturing with mobile phone cameras.

Speed: all post-capture processing should at most take a second or two (on the mobile device) before the 3D photo can be viewed and shared.

Compactness: the final representation should be easy to transmit and display on low-end devices.

Quality: depth discontinuities, and disocclusions should be handled well.

Intuitive Interaction: interactive viewing a 3D Photos requires should be both engaging and natural requiring no learning.

We built a practical implementation that satisfies these objectives and has been used to create more than 10 million 3D photos by end-users.

We leverage dual camera cell phones to capture color and depth image pairs. While the second image enables high quality stereo depth estimation, it does not help much with disocclusions because of the narrow camera baseline. We designed heuristic algorithms to detect major depth discontinuities, hallucinate occluded geometry, and inpaint colors on the newly formed geometry smoothly.

Our algorithms operate on a Layered Depth Image (LDI) representation [10], which is sparse (only stores structures that are really present in the scene), and provides inherent level- of-detail (nearby features are represented more densely than far ones), and can be converted into a compact textured mesh representation for transmission and rendering.

Our 3D Photos are received and consumed within the Facebook app on both iOS and Android devices as well as the Facebook website in a browser. They can also be consumed using a VR headset using a browser that supports WebVR, such as Oculus Browser on an Oculus Go or GearVR, or Firefox on a PC with an Oculus Rift. Each consumption platform has slightly different affordances (e.g., scrolling, gyro tracking, head motion) for consumption as described later.

2. Previous Work

The classic way to create 3D "tours into the picture" involves carefully annotating the depth of a picture manually, or semi-manually with the help of tools [5, 9].

More recent methods employ learning-based methods to estimate depth [6] and hallucinate novel views [2, 11, 1]. This is sometimes even done on-the-fly during rendering [7, 8]. A major advantage of "neural" re-rendering is that it does not require any pre-processing. However, it needs strong hardware at runtime to perform the network inference in real-time. Networks are often also only trained for specific scenes, and these methods do not guarantee that disocclusions are filled consistency from different viewing angles.

Stereo Magnification [12] synthesizes a multi-plane representation from the small-baseline stereo pair. It has the ability to hallucinate texture and structures in disocclusions, but the representation produces artifacts on sloped surfaces, and is, due to the excessive redundancy, memory and storage inefficient.

Our work leverages Layered Depth Images [10], similar to recent work [3, 4], but we are using more sophisticated heuristics for inpainting occlusions, and optimized the algorithms to perform end-to-end within 1-2 seconds on mobile devices.

3. 3D Photo Construction

The input to our algorithm is a color and depth image pair, typically captured with a dual camera cell phone. (We briefly discuss ways to derive depth from monocular imagery in future work.) Our processing algorithm consists of four stages, describes briefly for this extended abstract:

1. Pre-processing: We start by normalizing the depth range of the image, so each scene induces the same amount of screen-space parallax, independent of its real-life scale. Discontinuities in the depth map are the most salient feature, since they result in disocclusions when viewing. However, they are typically represented as soft edges in the depth map that spread over multiple pixels. We use image filters to clean the depth maps up and turn discontinuities into step edges that are precisely aligned with image edges, and reduce small scale features that would be difficult to represent faithfully.

2. Disocclusion: We next lift the depth map onto a LDI, at this point with just a single layer everywhere. We connect every pixel to its cardinal (left, right, top, bottom) neighbors, except across discontinuities. Then, we iteratively extend the *back*side discontinuities into the occluded region using a diffusion-based inpainting algorithm. We employ linear constraints to ensure that curve-like disocclusions continue straight into the occluded region. Iterative isotropic diffusion inpaints colors into the occluded regions. This algorithm is very fast and produces predictable, albeit smooth, results. (We also touch on future efforts later.)

3. Texture atlas: The next step is to decompose the LDI into a set of non-overlapping charts that can be packed into a texture atlas (Figure 1c). We use a *seed and grow* algorithm to produce these charts. Charts are padded for texture filtering.

4. Mesh generation: Finally, we convert the LDI geometry into a textured mesh. Since general mesh simplification algorithms are slow, we designed a custom algorithm that operates in the texture atlas domain and exploits the 2.5D nature of our data. We first simplify the chart boundaries into poly-lines. Then, we add evenly distributed interior points to capture internal depth variation. Finally, we generate a triangle mesh for each chart by converting it into a set of Y-monotone polygons and triangulating them independently. The result is represented as a *glTF* file for transmission.

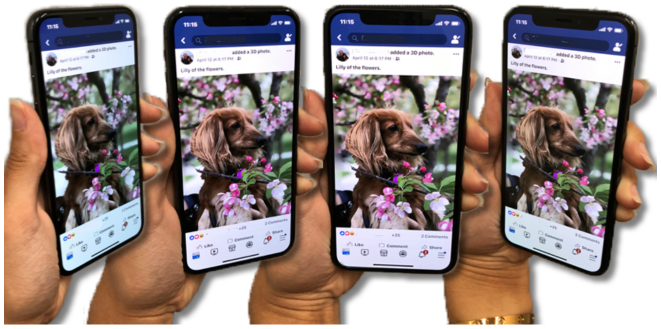
Figure 2. Rotating phone induces parallax.

Discussion: Our algorithm has been designed with performance in mind. This table summarizes the runtime on an iPhone X:

Pre-processing	100ms
Geometry inpainting	600ms
Color inpainting	500ms
Texture atlas generation	500ms
Mesh generation	100ms
Total	1800ms

4. 3D Photo Viewing

Without motion, a 3D Photo is just a 2D Photo. Fully experiencing the 3D format requires moving the virtual viewpoint to recreate the parallax one would see in the real world. We have designed interfaces for both Mobile devices and fixed browsers, as well as for heads-up viewing in VR where we also leverage stereo viewing.

1. Mobile and Browser:

On mobile devices there a number of possible affordances that can be mapped to virtual camera motion. These include scrolling in the application interface, IMUs such as the gyros to detect rotation, as well as finger touches.

After considerable user testing we finally ended up mapping scrolling behavior to both vertical rotation (about the horizontal axis) as well as dolling in and out (translation along the ”z” axis). This gives the illusion during scrolling the imagery from above to below, of the viewing point starting from below moving to above while simultaneously moving closer to the scene. We also added a small bit of horizontal rotation (about the vertical axis) mapped to scrolling. Although all rotation could be mapped to rotations of the device detected by the gyros, we found only mapping horizontal rotation to gyro motion to be most effective (Figure 2). In a web browser we substitute mouse motion for gyro rotation.

2. In Virtual Reality:

In VR, we have the additional advantage of being able to produce two offset images for each eye to produce stereo. This creates a significant feeling of immersion. 3D Photos are the only general immersive user-generated-content in VR.

We use threeJS (a Javascript 3D library) to render the scene to a WebGL context, and we use WebVR to render this context to a VR Device. The renderer queries the device parameters (eye buffer size and transforms), applying the information separately for the left and right eye views to produce a stereo image.

In addition to stereo we map head motion directly to virtual camera motion. In 6-DOF headsets, this is a one-to-one mapping. In 3-DOF (rotation only) we mimic head translation from rotation since rotating the head to the left, for example, also translates the eyes leftward.

One issue we solve is creating the model with limited absolute depth information. To minimize holes appearing in the model we create a frame around the model and constrain the viewing angles. The result appears like a 3D model popping out of a 2D frame.

5. Conclusions and Future Work

There are three key areas where we expect to apply learning algorithms to 3D Photo construction: predicting depth from monocular imagery, replace all heuristics, and hidden texture and structure in-painting.

By presenting a photographer with the means to capture a new type of media with no new effort beyond pressing a shutter button, and providing consumers intuitive interfaces for viewing 3D Photos, we have brought a new medium to millions of people by leveraging and enhancing both computer vision and computer graphics techniques.

References

[1] Inchang Choi, Orazio Gallo, Alejandro J. Troccoli, Min H. Kim, and Jan Kautz. Extreme view synthesis. 2018. 2

[2] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. Deepstereo: Learning to predict new views from the world’s imagery. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[3] Peter Hedman, Suhib Alsisan, Richard Szeliski, and Johannes Kopf. Casual 3d photography. *ACM Trans. Graph.*, 36(6):article no. 234, 2017. 1, 2

[4] Peter Hedman and Johannes Kopf. Instant 3d photography. *ACM Trans. Graph.*, 37(4):article no. 101, 2018. 2

[5] Youichi Horry, Ken-Ichi Anjyo, and Kiyoshi Arai. Tour into the picture: Using a spidery mesh interface to make animation from a single image. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH ’97, pages 225–232, 1997. 2

[6] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *Computer Vision and Pattern Recognition (CVPR)*, 2018. 2

[7] Ricardo Martin-Brualla, Rohit Pandey, Shuoran Yang, Pavel Pidlypenskyi, Jonathan Taylor, Julien Valentin, Sameh Khamis, Philip Davidson, Anastasia Tkach, Peter Lincoln,

Adarsh Kowdle, Christoph Rhemann, Dan B Goldman, Cem Keskin, Steve Seitz, Shahram Izadi, and Sean Fanello. Lookingood: Enhancing performance capture with real-time neural re-rendering. *ACM Trans. Graph.*, 37(6):article no. 255, 2018. 2

[8] Moustafa Meshry, Dan B Goldman, Sameh Khamis, Hugues Hoppe, Rohit Pandey, Noah Snavely, and Ricardo Martin-Brualla. Neural rerendering in the wild. *arXiv preprint*, 2019. 2

[9] Byong Mok Oh, Max Chen, Julie Dorsey, and Frédo Durand. Image-based modeling and photo editing. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pages 433–442, 2001. 2

[10] Jonathan Shade, Steven Gortler, Li-wei He, and Richard Szeliski. Layered depth images. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, pages 231–242, 1998. 2

[11] Pratul P. Srinivasan, Tongzhou Wang, Ashwin Sreelal, Ravi Ramamoorthi, and Ren Ng. Learning to synthesize a 4d RGBD light field from a single image. In *IEEE International Conference on Computer Vision, ICCV*, pages 2262–2270, 2017. 2

[12] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: Learning view synthesis using multiplane images. *ACM Trans. Graph.*, 37(4):article no. 65, 2018. 2

[13] C. Lawrence Zitnick, Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski. High-quality video view interpolation using a layered representation. *ACM Trans. Graph.*, 23(3):600–608, 2004. 1