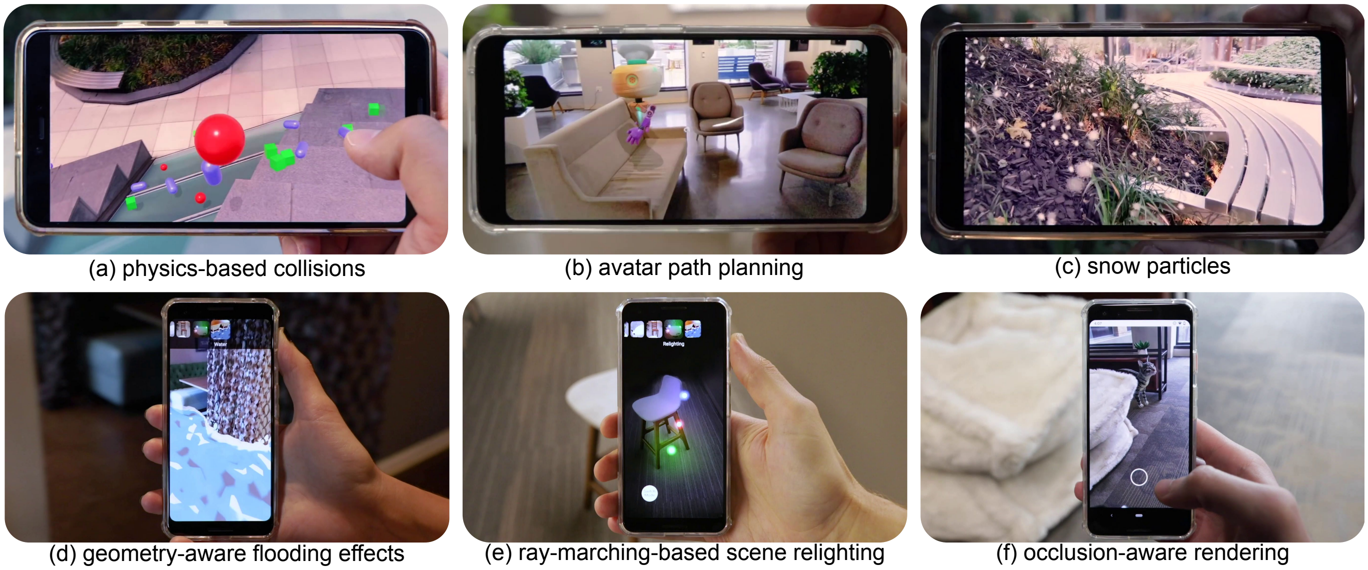
Experiencing Real-time 3D Interaction with Depth Maps for Mobile Augmented Reality in DepthLab

Ruofei Du, Eric Turner, Maksym Dzitsiuk, Luca Prasso, Ivo Duarte, Jason Dourgarian, Joao Afonso, Jose Pascoal, Josh Gladstone, Nuno Cruces, Shahram Izadi, Adarsh Kowdle, Konstantine Tsotsos, David Kim[†]
Google LLC



(a) physics-based collisions

(b) avatar path planning

(c) snow particles

(d) geometry-aware flooding effects

(e) ray-marching-based scene relighting

(f) occlusion-aware rendering

Figure 1. Real-time interactive components enabled by DepthLab: (a) virtual objects colliding with real stairs; (b) virtual avatar path planning and geometry-aware shadows; (c) AR snow effect; (d) virtual flooding effects bounded by physical walls; (e) scene relighting with three virtual point lights; (f) occlusion-aware rendering of a virtual cat behind the real bed. Please refer to the main paper [1] and the accompanying video for more results.

ABSTRACT

We demonstrate DepthLab [1], a playground for interactive augmented reality experiences leveraging the shape and depth of the physical environment on a mobile phone. Based on the ARCore Depth API, DepthLab encapsulates a variety of depth-based UI/UX paradigms, including geometry-aware rendering (occlusion, shadows, texture decals), surface interaction behaviors (physics, collision detection, avatar path planning), and visual effects (relighting, 3D-anchored focus and aperture effects, 3D photos). We have open-sourced our software at `https://github.com/googlesamples/arcore-depth-lab` to facilitate future research and development in depth-aware mobile AR experiences. With DepthLab, we aim to help mobile developers to effortlessly integrate depth into their AR experiences and amplify the expression of their creative vision.

Author Keywords

Depth map; interactive 3D graphics; real time; interaction; augmented reality; mobile AR; rendering; GPU; ARCore.

CCS Concepts

•**Human-centered computing** → **Mixed / augmented reality;** *User interface toolkits;*

INTRODUCTION

Real-time depth data is readily available on mobile phones with passive or active sensors and on VR/AR devices. However, the use of this rich data about our environment is underexplored and merely leveraged in mainstream AR applications. In this demonstration paper, we present DepthLab [1], an opensourced library based on ARCore Depth API [2] that encapsulates a variety of real-time UI/UX features for depth, including geometry-aware rendering and physics simulation, surface interaction behaviors, and visual effects. Our goal is to bring these advanced features to mobile AR experiences without relying on dedicated sensors or the need for computationally expensive surface reconstruction.

[†] Corresponding author: kidavid@google.com.

UIST '20 Adjunct, October 20–23, 2020, Virtual Event, USA
© 2020 Copyright is held by the author/owner(s).
ACM ISBN 978-1-4503-7514-6/20/10.
http://dx.doi.org/10.1145/3379350.3416136

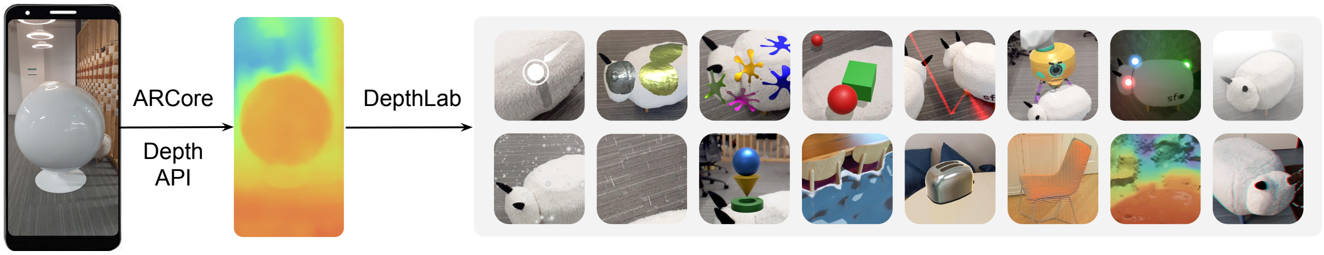


Figure 2. A high-level overview of DepthLab. We process the raw depth map from ARCore Depth API and provide customizable and self-contained components such as a 3D cursor, geometry-aware collision, and screen-space relighting. The DepthLab library aims to accelerate mobile app developers to build more photo-realistic and interactive AR applications.

System Overview

DepthLab enables users to interact with a seamless blend of the physical environment and virtual renderings. To achieve this, we architect and implement a set of data structures and real-time algorithms for mobile AR developers.

The depth data is typically stored in a low-resolution depth buffer (160 × 120 in our examples[1]), which is a perspective camera image that contains a depth value instead of color in each pixel. We generate three categories of data structures: (1) **Depth array** stores depth in a 2D array of a landscape image with 16-bit integers on the CPU. (2) **Depth mesh** is a real-time triangulated mesh generated for each depth map on both CPU and GPU. (3) **Depth texture** is copied to the GPU from the depth array for per-pixel depth use cases in each frame.

Based on the data structures, we classify our DepthLab components into three categories: (1) **Localized depth** uses the depth array to operate on a small number of points directly on the CPU. It is useful for computing physical measurements, estimating normal vectors, and automatically navigating virtual avatars for AR games. (2) **Surface depth** leverages the CPU or compute shaders on the GPU to create and update depth meshes in real time, thus enabling collision, physics, texture decal, geometry-aware shadows, etc. (3) **Dense depth** is copied to a GPU texture and is used for rendering depth-aware effects, including relighting, 3D-anchored focus and aperture, and screen-space occlusion effects.

Demonstration and Applications

In our demonstration, the audience will be able to watch a live stream by the presenters showcasing DepthLab on Pixel phones. In addition, visitors will be instructed to download DepthLab from Google Play Store[2].

During the demonstration, the audience will experience the following demos: (1) **Oriented Reticle** uses depth hit testing to obtain the raycasted 3D position and surface normal of a raycasted screen point. (2) **Material Wrap** allows the user to change the material of real-world surfaces through touch. (3) **Color Balloons** uses the Oriented Reticle and the depth mesh in placing a surface-aligned texture decal within the physical environment. (4) **Collider** uses screen-space depth meshes to enable collisions between Unity's rigid-body objects and the physical environment. (5) **Laser Beam** allows the user to shoot a slowly moving laser beam by touching anywhere on the screen. (6) **Avatar Path Planning** allows an AR character follows user-set waypoints while staying close to the surface of an uneven terrain. This scene uses raycasting and depth lookups on the CPU to calculate a 3D point on the surface of the terrain. (7) **Relighting** uses the GPU depth texture to computationally re-light the physical environment through the AR camera. Areas of the physical environment close to the artificial light sources are lit, while areas farther away are darkened. (8) **Fog** adds a virtual fog layer on the physical environment. Close objects will be more visible than objects further away. (9) **Snow** uses the GPU depth texture to compute collisions between snow particles, the physical environment, and the orientation of each snowflake. (10) **Rain** uses the GPU depth texture to compute collisions between rain particles and the physical environment. (11) **Focus and Aperture Effect** contains a simulated Bokeh-like fragment-shader effect. This blurs the regions of the AR view that are not at the user-defined focus distance. The focus anchor is set in the physical environment by touching the screen. The focus anchor is a 3D point that is locked to the environment and always in focus. (12) **Water** uses a modified GPU occlusion shader to create a flooding effect with artificial water in the physical environment. (14) **Object Placement** uses depth lookups on the CPU to test collisions between the vertices of virtual objects and the physical environment. (15) **Point Cloud** computes a point cloud on the CPU using the depth array. Press the Update button to compute a point cloud based on the latest depth data. (16) **Depth Mesh** renders a template mesh created once on the GPU as a regular grid of triangles. The GPU shader displaces each vertex of the regular grid based on the reprojection of the depth values provided by the GPU depth texture. (17) **3D Photo** uses depth meshes and project a cached camera image onto a frozen template mesh. As the virtual camera rotates, it create an animated stereo photo effects.

Conclusion

We demonstrate DepthLab, an interactive depth library that aims to empower mobile AR designers and developers to more realistically interact with the physical world using virtual content. We believe our contributions will inspire the next generation of AR applications, where scene-aware interactions, enabled by accurate 3D information, are the key to seamless blending of the virtual and the real world.

[1] The resolution may be different depending on phone models.

[2] ARCore Depth Lab: https://play.google.com/store/apps/details?id=com.google.ar.unity.arcore_depth_lab

REFERENCES

[1] Ruofei Du, Eric Turner, Max Dzitsiuk, Luca Prasso, Ivo Duarte, Jason Dourgarian, Joao Afonso, Jose Pascoal, Josh Gladstone, Nuno Cruces, Shahram Izadi, Adarsh Kowdle, Konstantine Tsotsos, and David Kim. 2020. DepthLab: Real-Time 3D Interaction With Depth Maps for Mobile Augmented Reality. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology (UIST)*. ACM, 15. DOI: **http://dx.doi.org/10.1145/3379337.3415881**

[2] Julien Valentin, Adarsh Kowdle, Jonathan T. Barron, Neal Wadhwa, Max Dzitsiuk, Michael Schoenberg, Vivek Verma, Ambrus Csaszar, Eric Turner, Ivan Dryanovski, Joao Afonso, Jose Pascoal, Konstantine Tsotsos, Mira Leung, Mirko Schmidt, Onur Guleryuz, Sameh Khamis, Vladimir Tankovitch, Sean Fanello, Shahram Izadi, and Christoph Rhemann. 2018. Depth From Motion for Smartphone AR. *ACM Trans. Graph* 37, 6, Article 193 (2018), 193:1–193:19 pages. DOI: **http://dx.doi.org/10.1145/3272127.3275041**