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

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 | ACM UIST 2020

Introduction

Mobile Augmented Reality

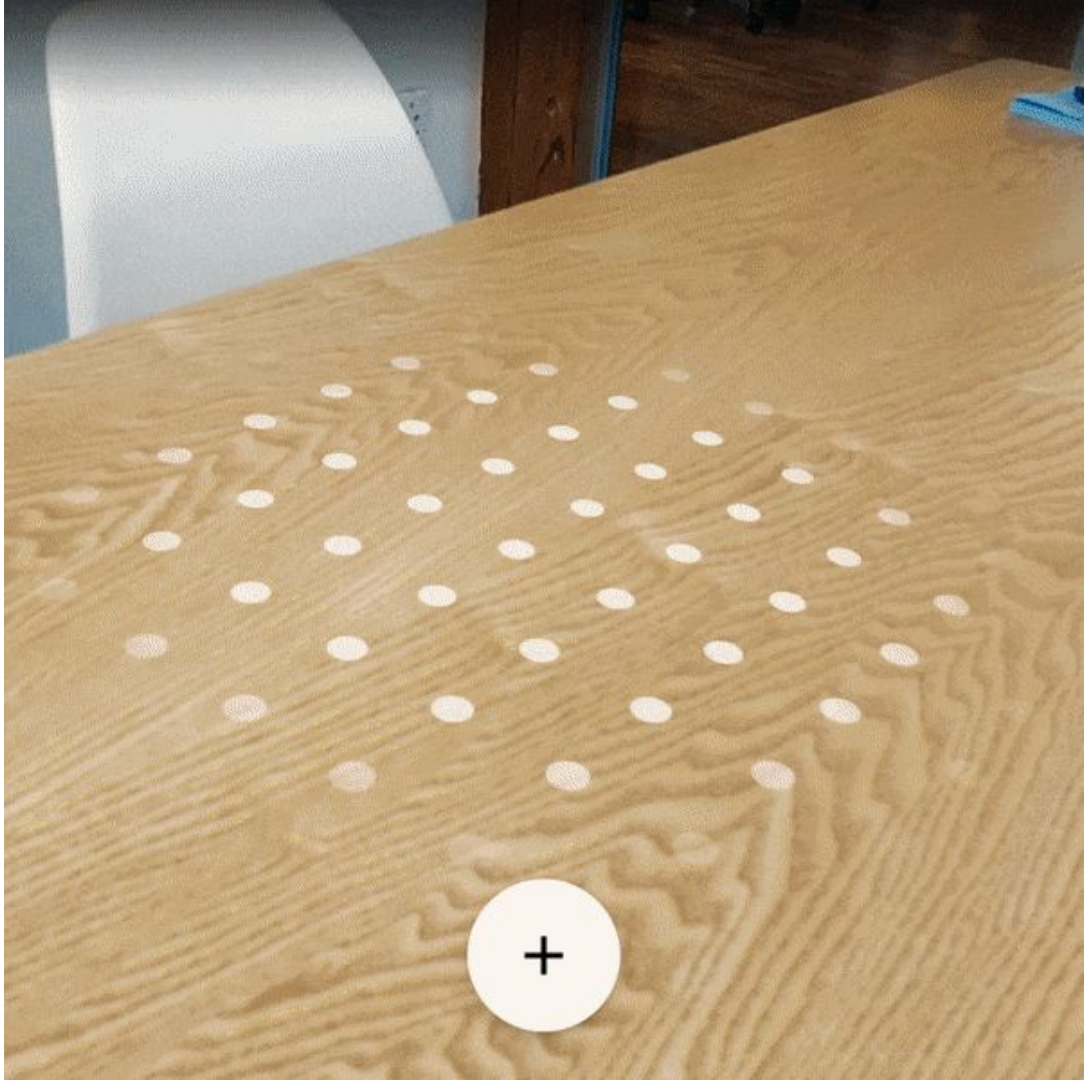
Introduction

Google's ARCore

Introduction

Google's ARCore

ARCore



Introduction

Mobile Augmented Reality

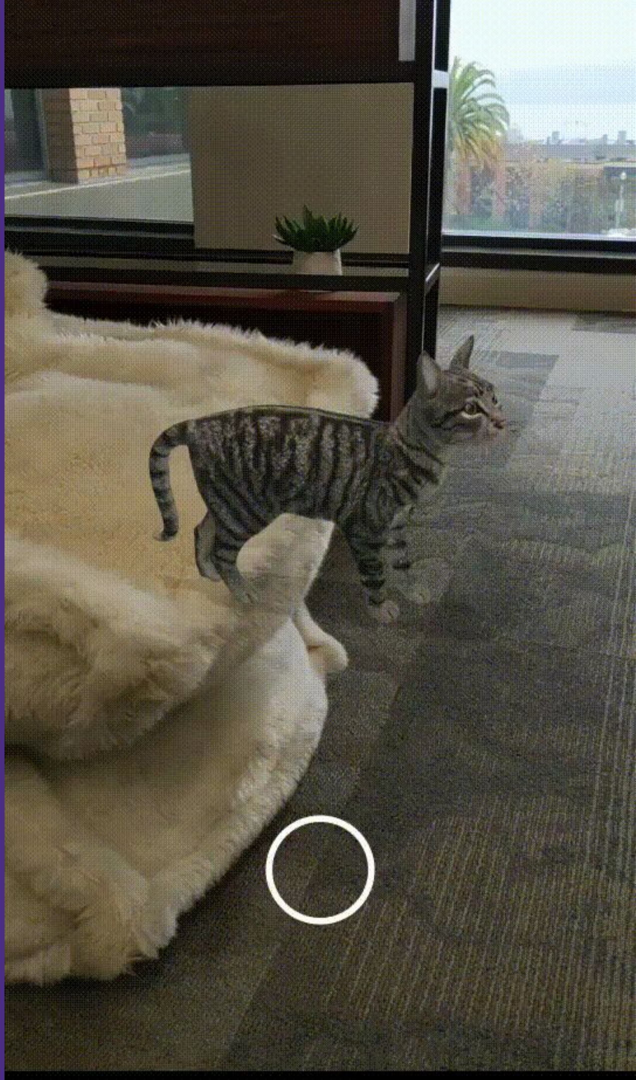
Is direct placement and rendering of 3D objects sufficient for realistic AR experiences?

Not always!

Introduction

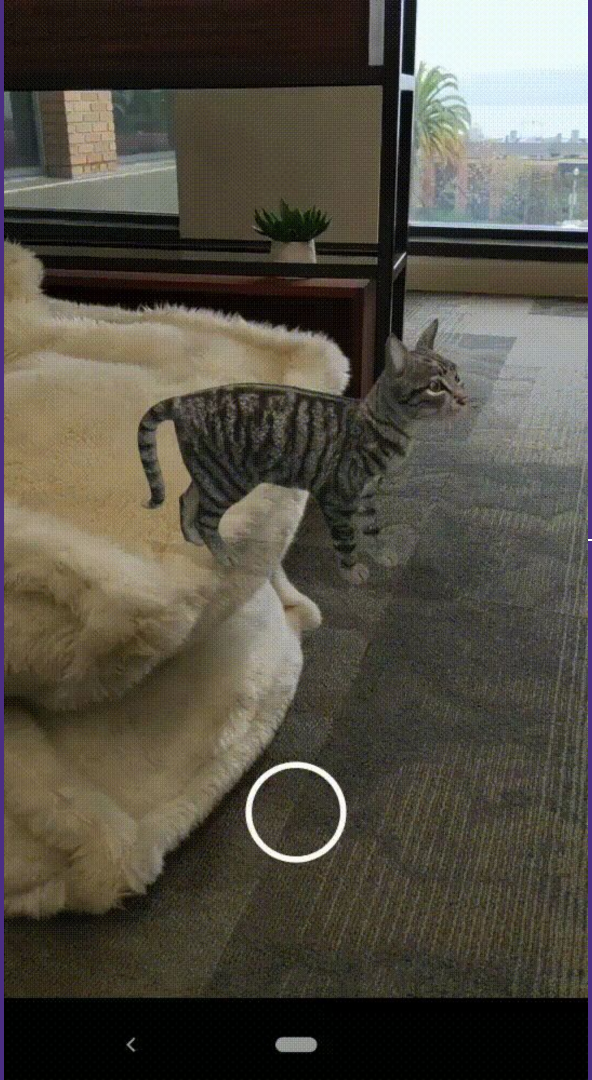
Depth Lab

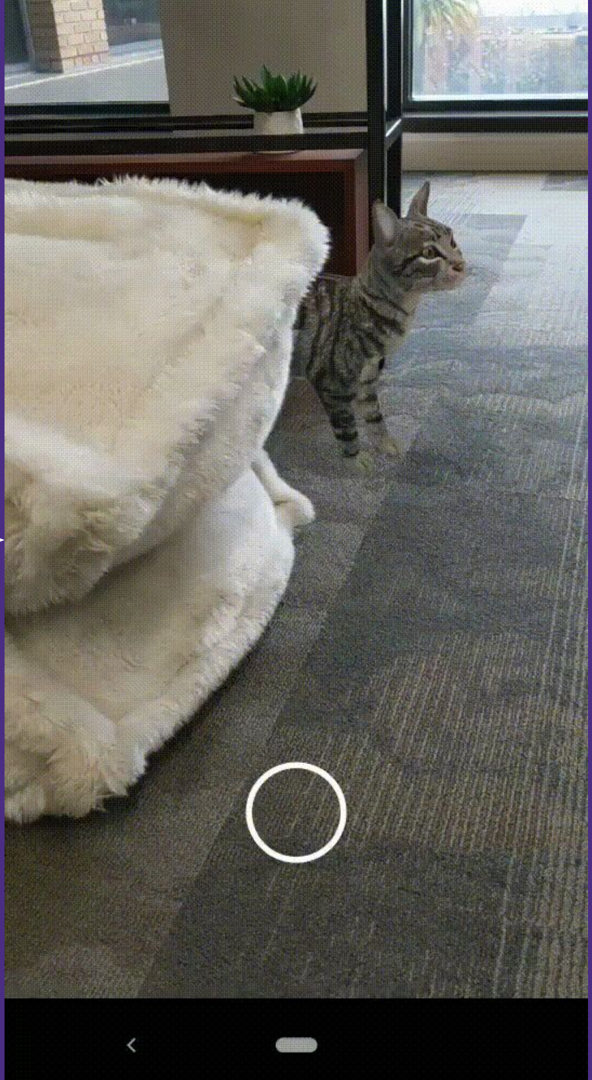
Virtual content looks like it's *"pasted on the screen"* rather than *"in the world"*!



Introduction

Motivation





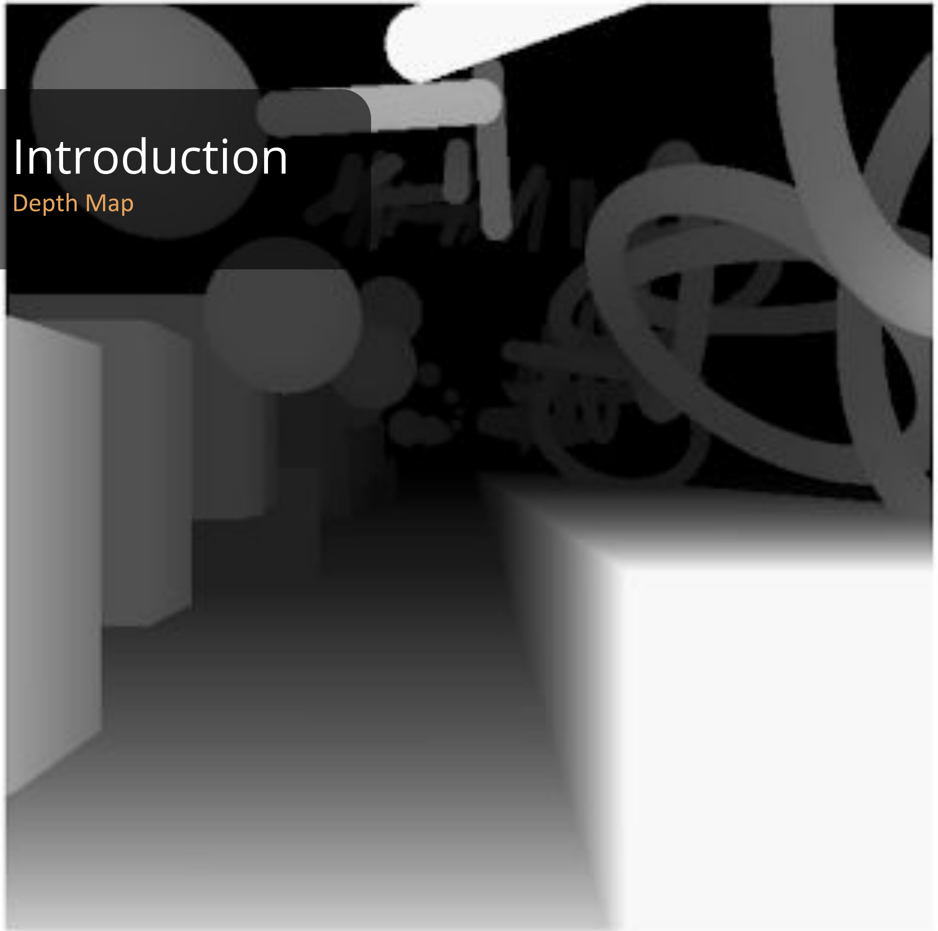
Introduction

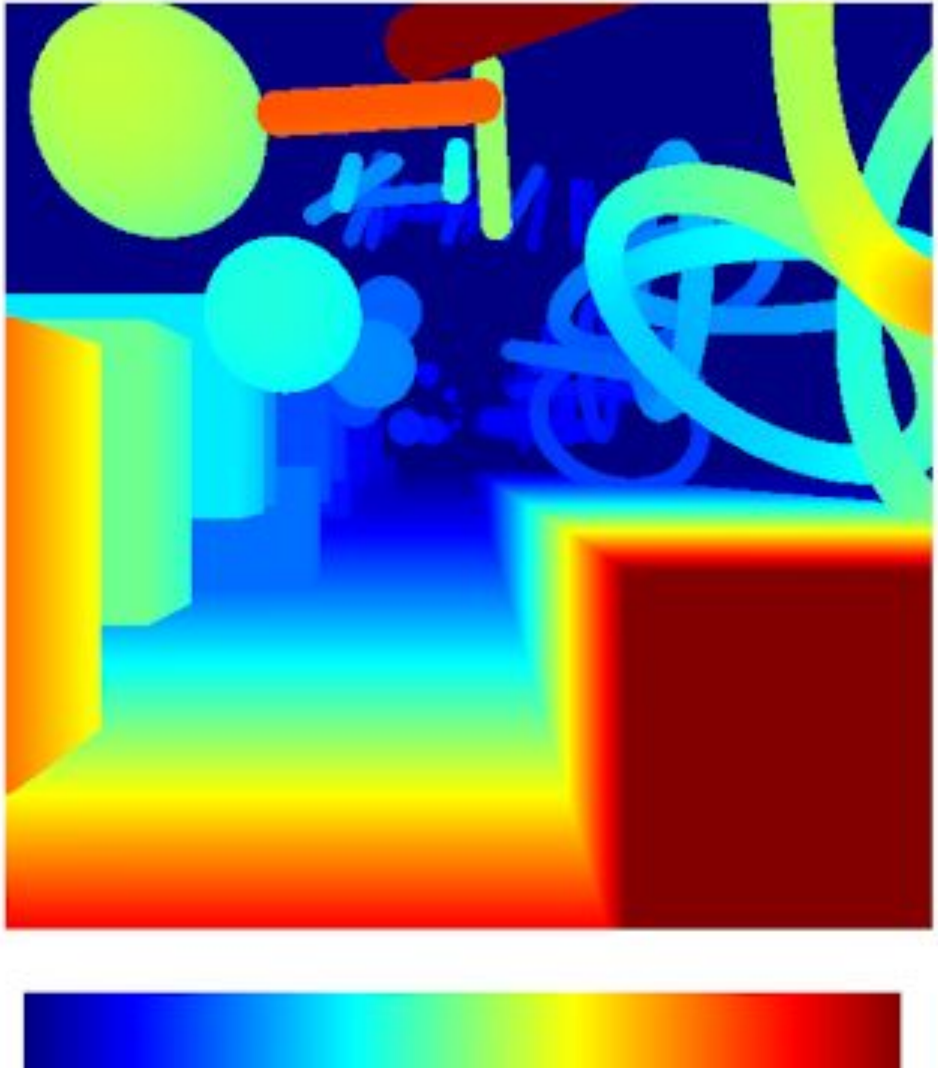
Motivation

How can we bring these advanced features to mobile AR experiences without relying on dedicated sensors or the need for computationally expensive surface reconstruction?

Introduction

Depth Map





Introduction

Depth Lab

Google

- *Pixel 2, Pixel 2 XL, Pixel 3, Pixel 3 XL, Pixel 3a, Pixel 3a XL, Pixel 4, Pixel 4 XL*

Huawei

- *Honor 10, Honor V20, Mate 20 Lite, Mate 20, Mate 20 X, Nova 3, Nova 4, P20, P30, P30 Pro*

LG

- *G8X ThinQ, V35 ThinQ, V50S ThinQ, V60 ThinQ 5G*

OnePlus

- *OnePlus 6, OnePlus 6T, OnePlus 7, OnePlus 7 Pro, OnePlus 7 Pro 5G, OnePlus 7T, OnePlus 7T Pro*

Oppo

- *Reno Ace*

Samsung

- *Galaxy A80, Galaxy Note8, Galaxy Note9, Galaxy Note10, Galaxy Note10 5G, Galaxy Note10+, Galaxy Note10+ 5G, Galaxy S8, Galaxy S8+, Galaxy S9, Galaxy S9+, Galaxy S10e, Galaxy S10, Galaxy S10+, Galaxy S10 5G, Galaxy S20, Galaxy S20+ 5G, Galaxy S20 Ultra 5G*

Sony

- *Xperia XZ2, Xperia XZ2 Compact, Xperia XZ2 Premium, Xperia XZ3*

Xiaomi

- *Pocophone F1*

And growing…

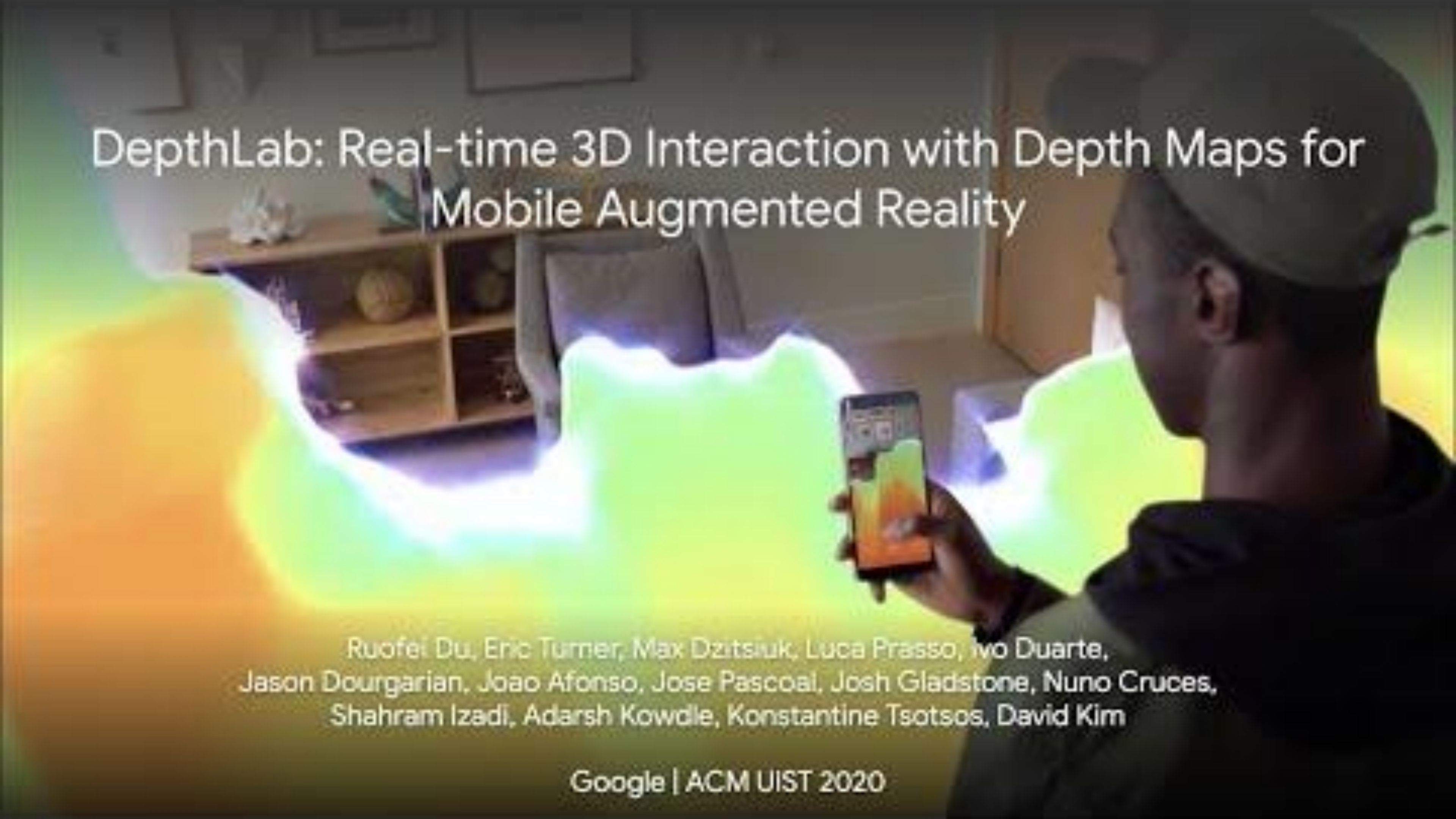
https://developers.google.com/ar/discover/supported-devices

Is there *more* to realism than occlusion?

Surface interaction?

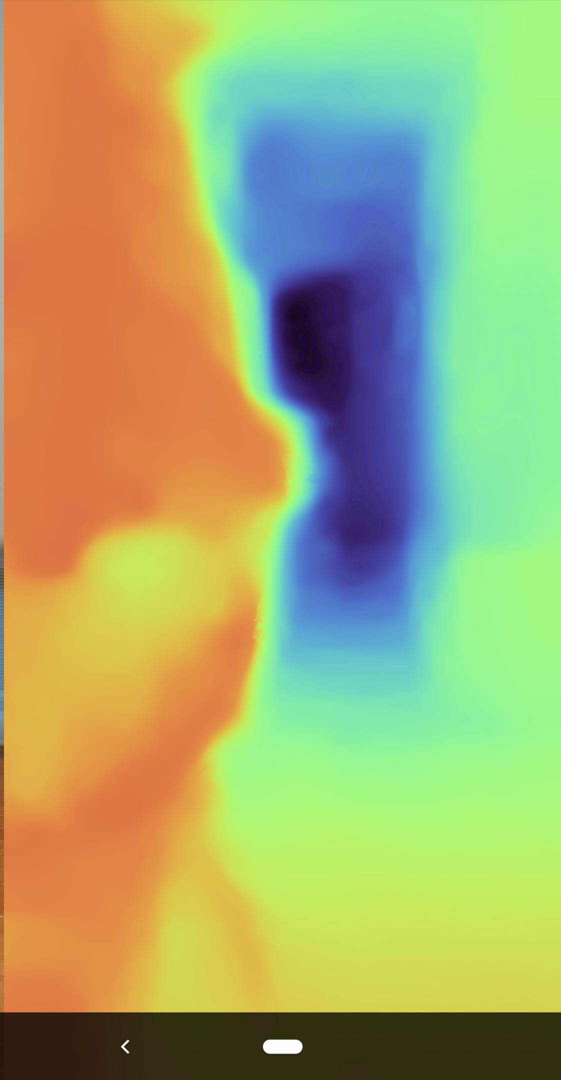
Realistic Physics?

Path Planning?


DepthLab: Real-time 3D Interaction with Depth Maps for Mobile Augmented Reality
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, David Kim
Google | ACM UIST 2020

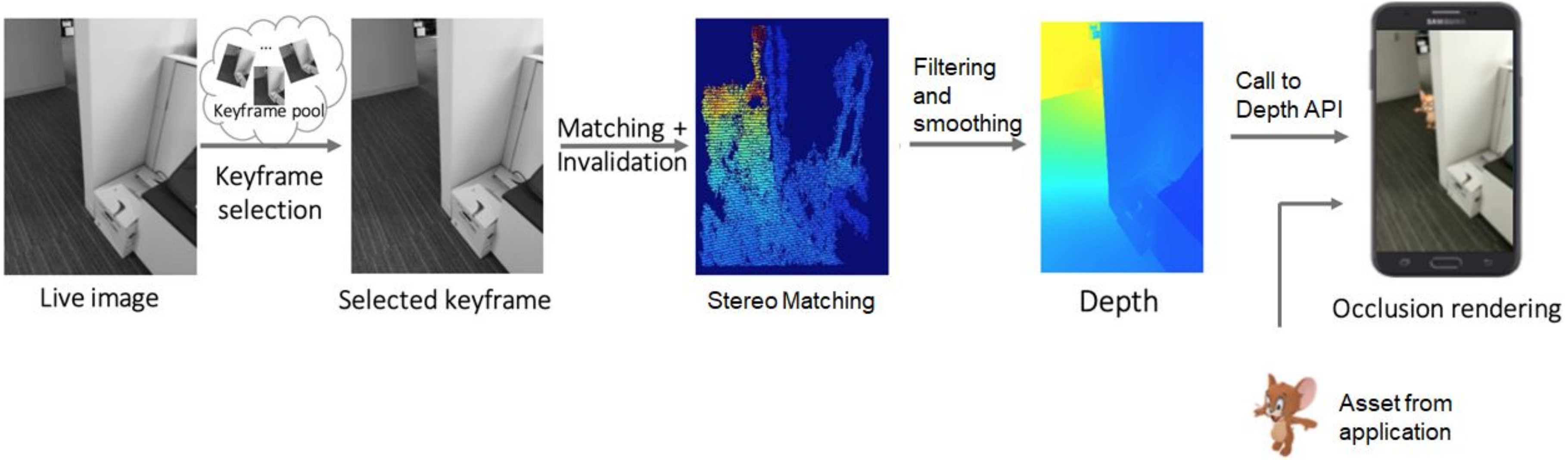
Introduction

Depth Lab



Related Work

Valentin et al.



[Depth from Motion for Smartphone AR (SIGGRAPH Asia 2018)]

Introduction
Depth Lab

Introduction

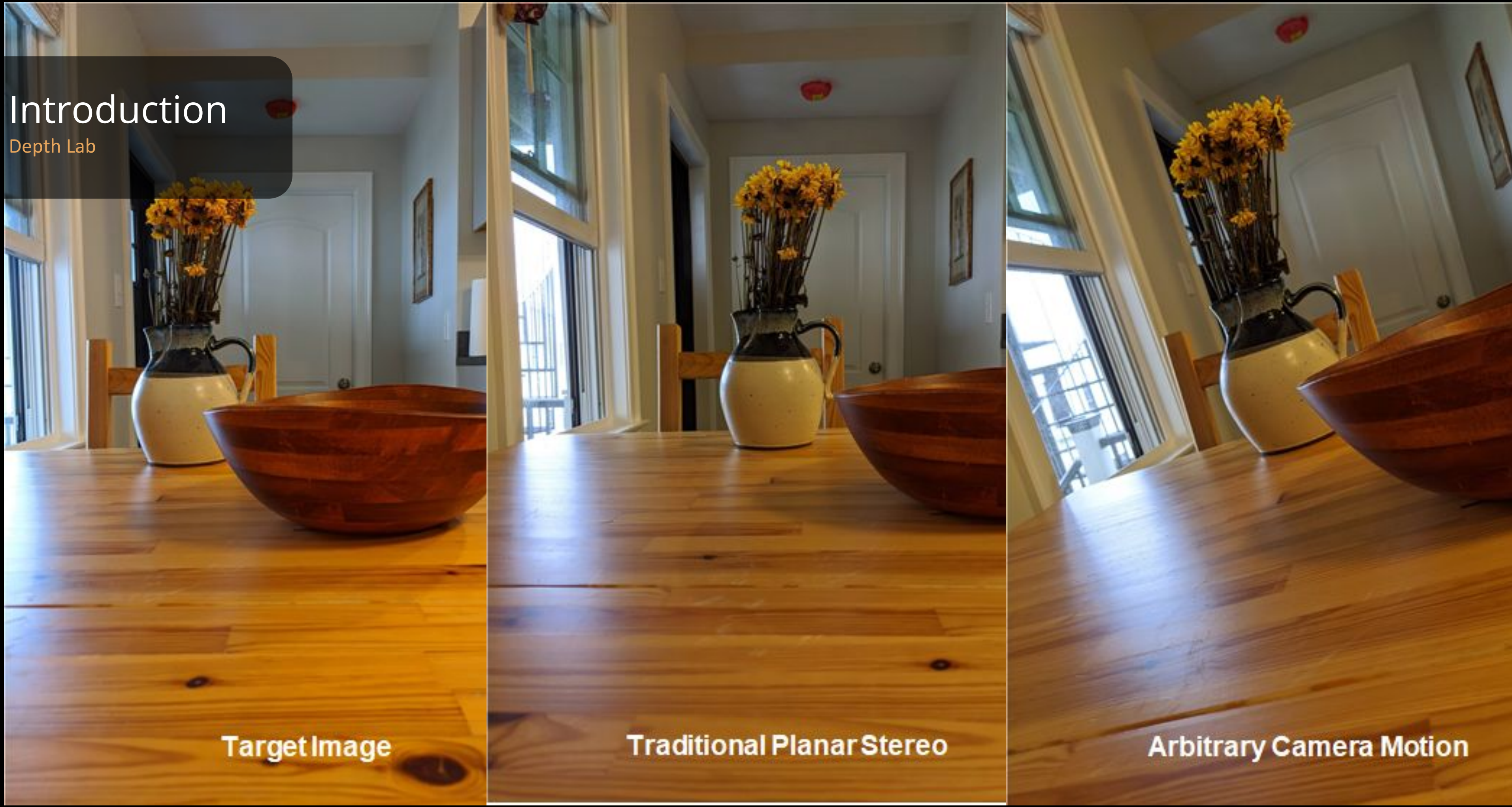
Depth Lab

Introduction

Depth Generation

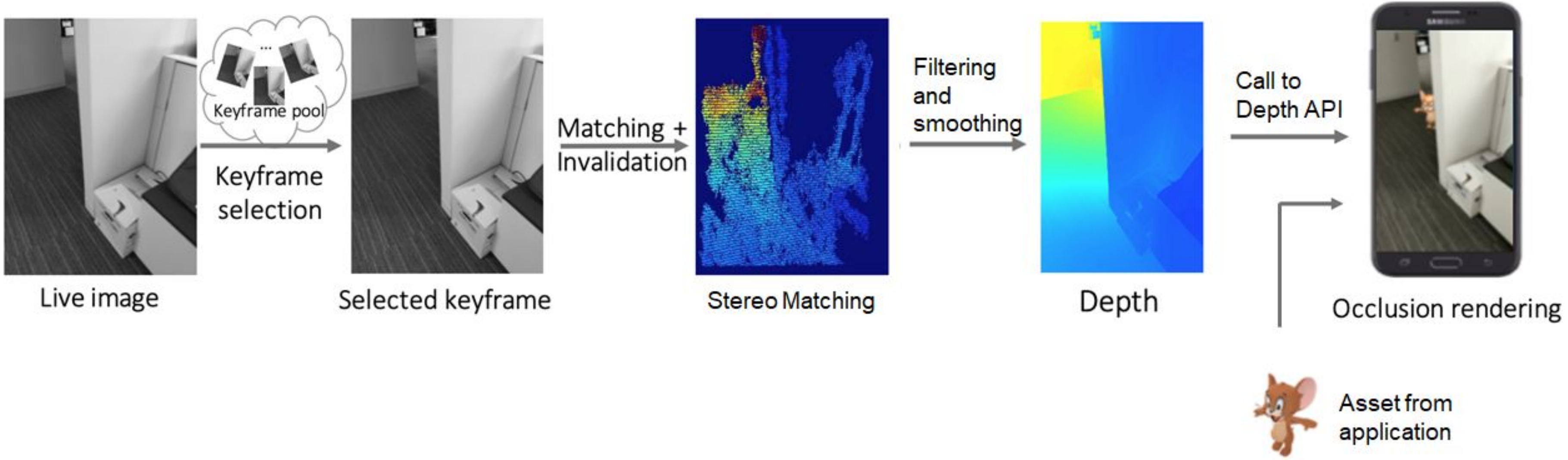
Introduction

Depth Lab



Related Work

Valentin et al.



Depth from Motion for Smartphone AR (SIGGRAPH Asia 2018)

Introduction

Depth Lab



Introduction

Depth Lab

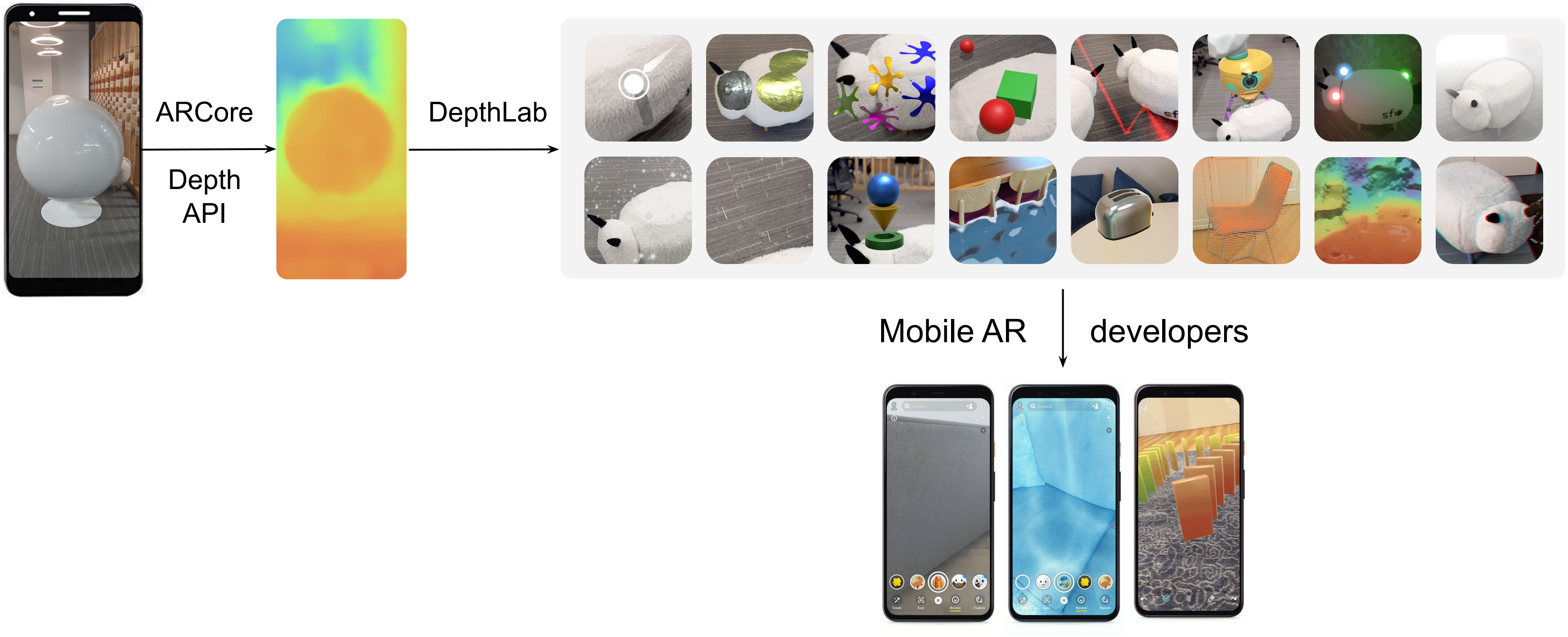
Up to 8 meters, with
the best within 0.5m to 5m

Motivation

Gap from raw depth to applications

Introduction

Depth Lab



Design Process

3 Brainstorming Sessions

3 brainstorming sessions
18 participants
39 aggregated ideas

Design Process

3 Brainstorming Sessions

Supplementary Material for DepthLab: Real-time 3D Interaction with Depth Maps for Mobile Augmented Reality

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

GEOMETRY-AWARE AR FEATURES

In this section, we list all ideas from our brainstorming sessions and discuss their depth representation requirements, use cases, and whether each is implemented in DepthLab [5]. Note that ideas 9, 21, 24, 25 are not available as open source code yet, but can be easily reproduced with the provided algorithms.

Depth Representation Requirement: Localized Depth

Figure 1. Implementation examples of geometry-aware AR features 1–5 with localized depth use cases. Please refer to the supplementary video for live demonstration.

1. 3D oriented cursor: Render a 3D cursor centered in the screen center. The 3D cursor should change its orientation and scale according to the surface normal and distance when moving along physical surfaces. Implemented in DepthLab: Yes.

4. Avatar path planning: Navigate a virtual object to move naturally between two points in physical environments. Implemented in DepthLab: Yes.

5. Collision-aware placement: Test if a virtual object's volume collides with observed environment surfaces. Implemented in DepthLab: Yes.

Depth Representation Requirement: Surface Depth

Figure 2. Implementation examples of geometry-aware AR features 6–9.

6. Virtual shadows: Render geometry-aware shadows [11] that are cast onto physical surfaces. The shadow may be ...

8. Physical simulation: Simulate physical phenomena for augmented reality objects, e.g. collision. Implemented in DepthLab: Yes.

9. AR graffiti: Allow the user to touch on the screen and sketch/spray/paint virtual drawings onto physical objects. Implemented in DepthLab: Yes.

Figure 3. Implementation examples of geometry-aware AR features 15–25 with dense depth.

19. Rain effects: Similiar in behavior to the snow effect, the rain particles should also splat on the surface using the estimated normal vector from the localized depth. Implemented in DepthLab: Yes.

20. Fog effects: Render screen-space post-processing effects, where far objects are overlaid with thicker fog. The user may interactively adjust the fog intensity in real time. Implemented in DepthLab: Yes.

21. Edge highlighting: Highlight the edges of the observed environment according to the depth map. Unlike edge detection in a color image, highlighting depth edges may offer a clean segmentation of physical objects regardless of their texture. Implemented in DepthLab: Yes.

22. Depth-based segmentation: Segment the foreground, background, or objects between a certain range of depth values from the color image. It may be useful for teleconferencing tasks. Implemented in DepthLab: Yes.

23. False-color visualization and animated transition effects: Visualize the depth map based on a specific transfer function and animate the transition ...

35. Enable multitouch on surfaces [12]: User may annotate sticky notes and papers with a pen and "program" them to control smart lights, music, and other digital functions of the environment.

36. Person capture: Enable self-scanning with the frontal camera [1] and teleconference with the rear camera.

37. AR board game: Design an AR-based board game [2] that overlays digital assets upon physical cards with aware of users' gestures and actions.

38. Interactive surface editing: Apply simple 3D distortion (pinch, twist, taper, bend) to captured colored voxels of the physical environment [4].

39. Interactive music experience: Design in-air instruments (guitar, piano) with dynamic gesture recognition [8]. Virtual targets are placed in 3D space, such as a drum set, big piano keys, etc. Upon contact detection, the app plays a sound.

REFERENCES

[1] Karan Ahuja, Chris Harrison, Mayank Goel, and Robert Xiao. 2019. MeCap: Whole-Body Digitization for Low-Cost VR/AR Headsets. In Proceedings of the 32Nd Annual ACM Symposium on User Interface Software and Technology (UIST '19). ACM, ACM, New York, NY, USA, 453–462. DOI: http://dx.doi.org/10.1145/3332165.3347889

[2] Troels L Andersen, Sune Kristensen, Bjørn W Nielsen, and Kaj Grønbæk. 2004. Designing an augmented reality board game with children: the battleboard 3D experience. In Proceedings of the 2004 conference on Interaction design and children: building a community. 137–138.

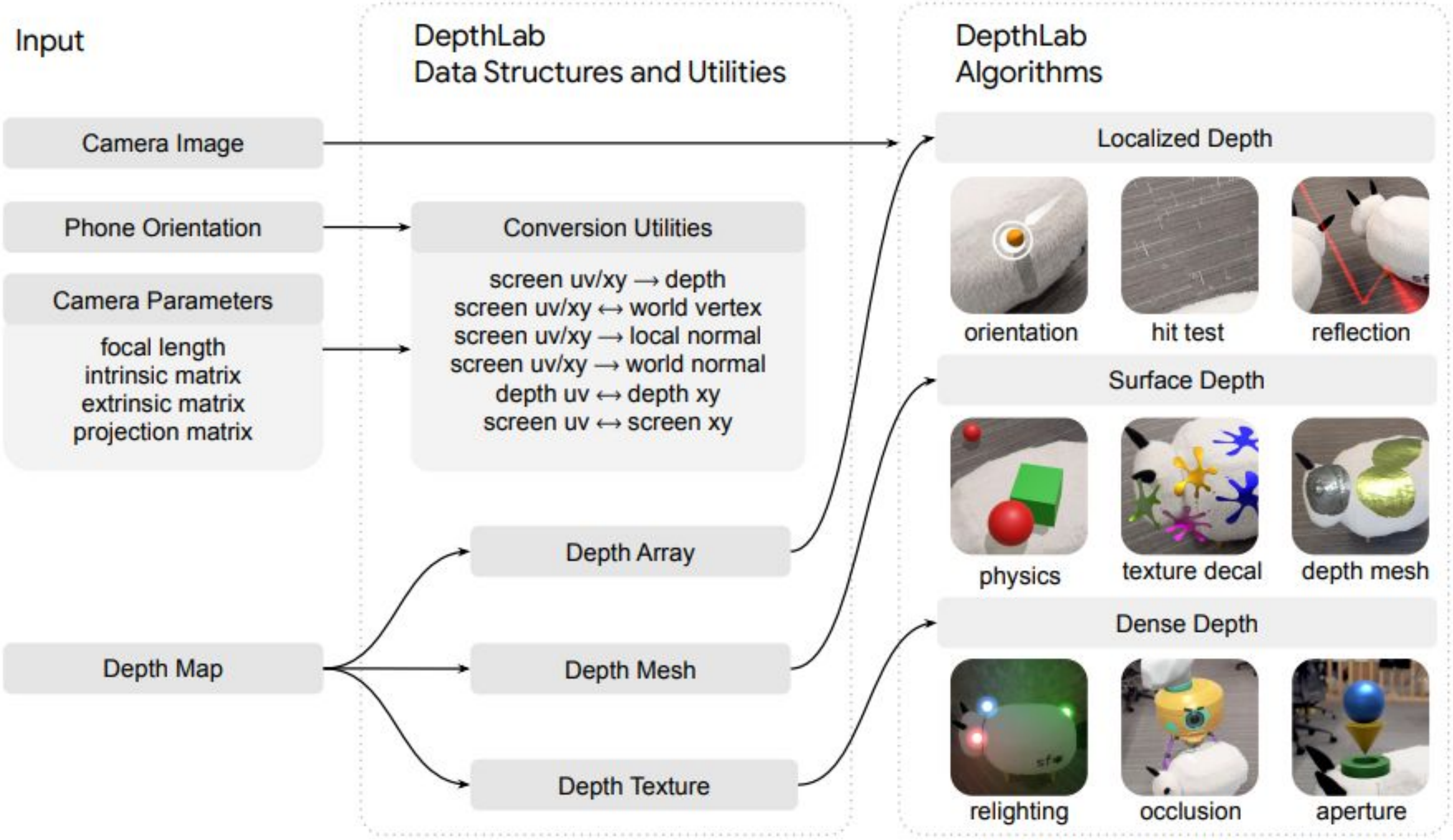
[3] Sujal Bista, Icaro Lins Leitao da Cunha, and Amitabh Varshney. 2017. Kinetic Depth Images: Flexible Generation of Depth Perception. The Visual Computer 33, 10 (01 October 2017), 1357–1369.

[4] Ming Chuang and Michael Kazhdan. 2011. Interactive and anisotropic geometry processing using the screened Poisson equation. In ACM SIGGRAPH 2011 papers. 1–10.

[5] 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 ...

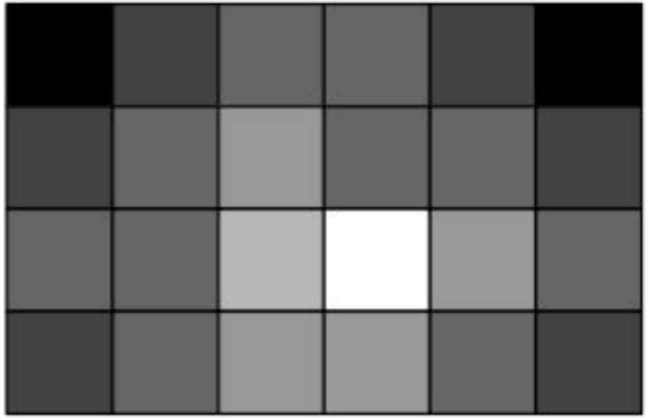
System

Architecture overview



Data Structure

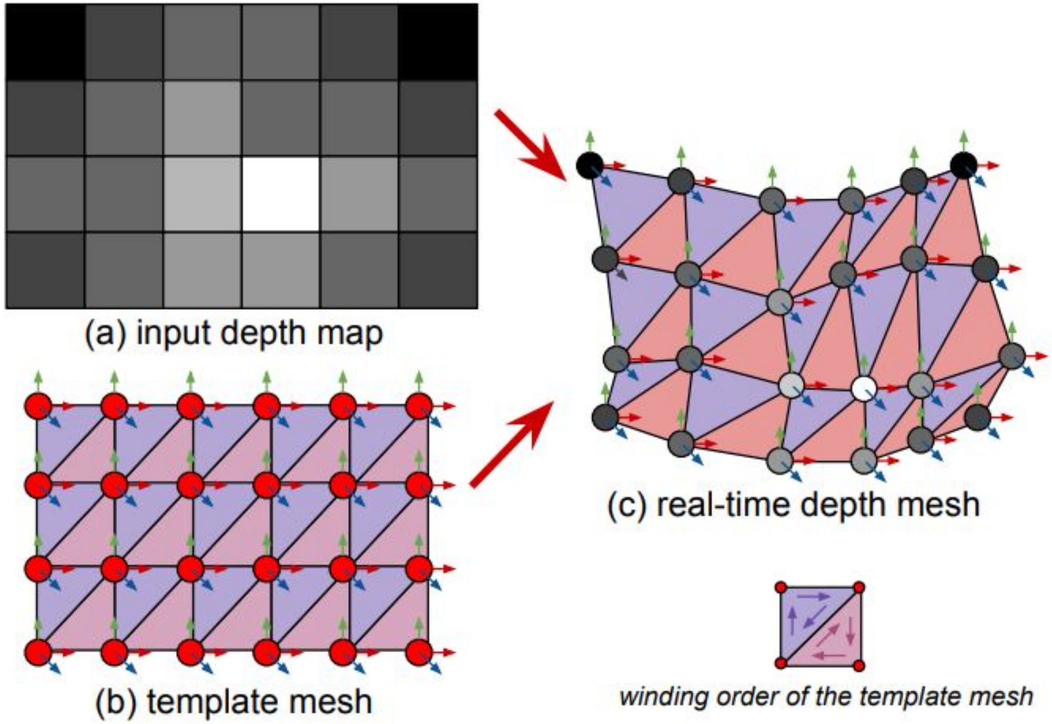
Depth Array



2D array (160x120 and above) of 16-bit integers

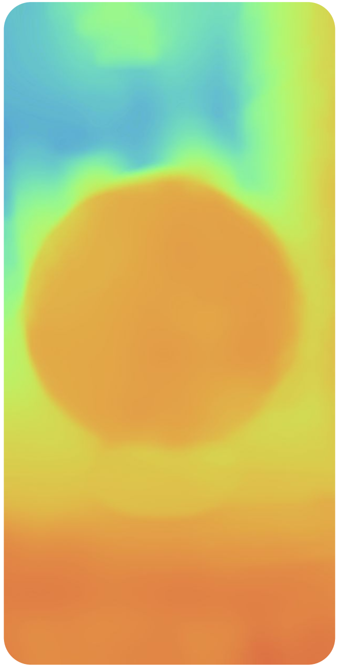
Data Structure

Depth Mesh



Data Structure

Depth Texture



System
Architecture

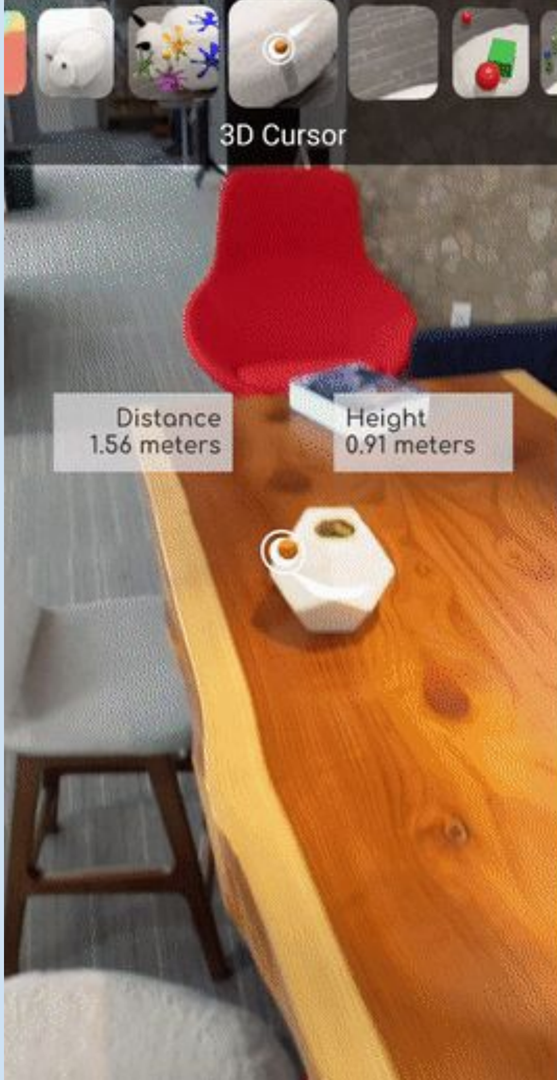
	Localized Depth	Surface Depth	Dense Depth
CPU	✓	✓	✗ (non-real-time)
GPU	N/A	✓ (compute shader)	✓ (fragment shader)
Prerequisite	point projection normal estimation	depth mesh triplanar mapping	anti-aliasing multi-pass rendering
Data Structure	depth array	depth mesh	depth texture
Example Use Cases	physical measure oriented 3D cursor path planning	collision & physics virtual shadows texture decals	scene relighting aperture effects occluded objects

Localized Depth

Coordinate System Conversion

Conversion Utilities

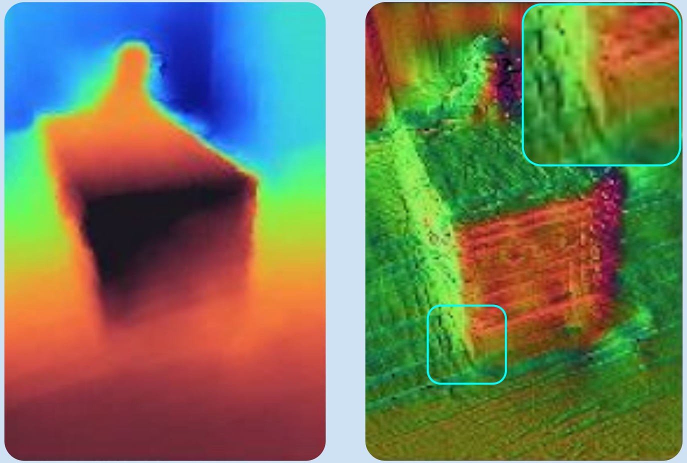
screen uv/xy ⟶ depth
screen uv/xy ⟷ world vertex
screen uv/xy ⟶ local normal
screen uv/xy ⟶ world normal
depth uv ⟷ depth xy
screen uv ⟷ screen xy



Localized Depth

Normal Estimation

$$\mathbf{n}_\mathbf{p} = \left(\mathbf{v}_\mathbf{p} - \mathbf{v}_{\mathbf{p}+(1,0)}\right) \times \left(\mathbf{v}_\mathbf{p} - \mathbf{v}_{\mathbf{p}+(0,1)}\right)$$



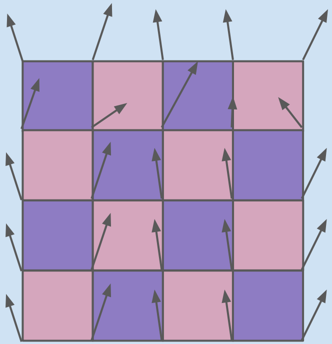
Localized Depth

Normal Estimation

Algorithm 1: Estimation of the Normal Vector of a Screen Point in DepthLab.

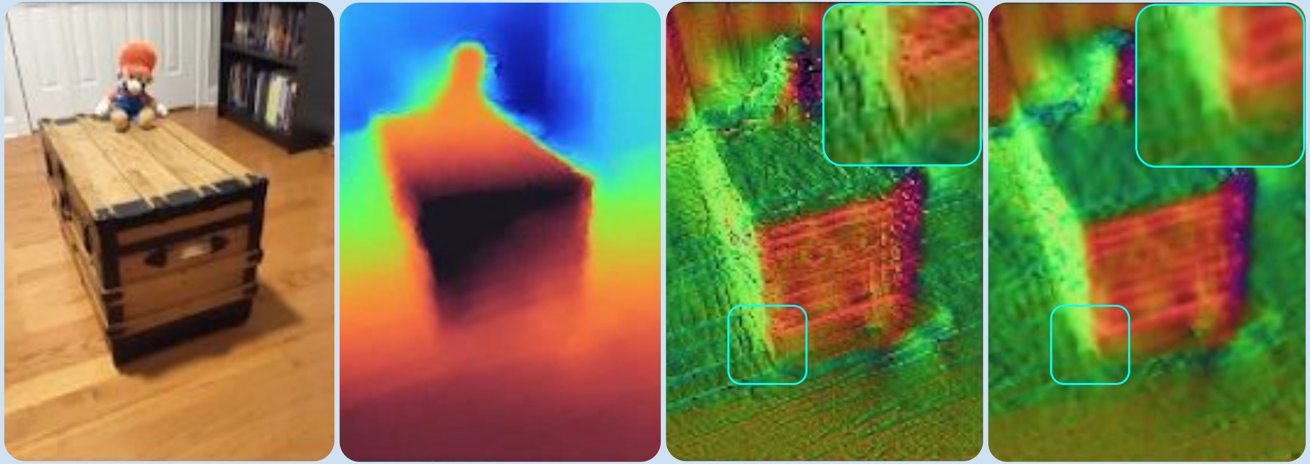
Input : A screen point $\mathbf{p} \leftarrow (x, y)$ and focal length f.
Output : The estimated normal vector $\mathbf{n}$.

Set the sample radius: $r \leftarrow 2$ pixels.
Initialize the counts along two axes: $c_X \leftarrow 0, c_Y \leftarrow 0$.
Initialize the correlation along two axes: $\rho_X \leftarrow 0, \rho_Y \leftarrow 0$.
for $\Delta x \in [-r, r]$ **do**
  for $\Delta y \in [-r, r]$ **do**
    Continue if $\Delta x = 0$ **and** $\Delta y = 0$.
    Set neighbor's coordinates: $\mathbf{q} \leftarrow [x + \Delta x, y + \Delta y]$.
    Set $\mathbf{q}$'s distance in depth: $d_{\mathbf{pq}} \leftarrow \|\mathbf{D}(\mathbf{p}), \mathbf{D}(\mathbf{q})\|$.
    Continue if $d_{\mathbf{pq}} = 0$.
    if $\Delta x \neq 0$ **then**
      $c_X \leftarrow c_X + 1$.
      $\rho_X \leftarrow \rho_X + d_{\mathbf{pq}} / \Delta x$.
    end
    if $\Delta y \neq 0$ **then**
      $c_Y \leftarrow c_Y + 1$.
      $\rho_Y \leftarrow \rho_Y + d_{\mathbf{pq}} / \Delta y$.
    end
  end
end
Set pixel size: $\lambda \leftarrow \frac{\mathbf{D}(\mathbf{p})}{f}$.
return the normal vector $\mathbf{n}$: $\left(-\frac{\rho_Y}{\lambda c_Y}, -\frac{\rho_X}{\lambda c_X}, -1\right)$.



Localized Depth

Normal Estimation

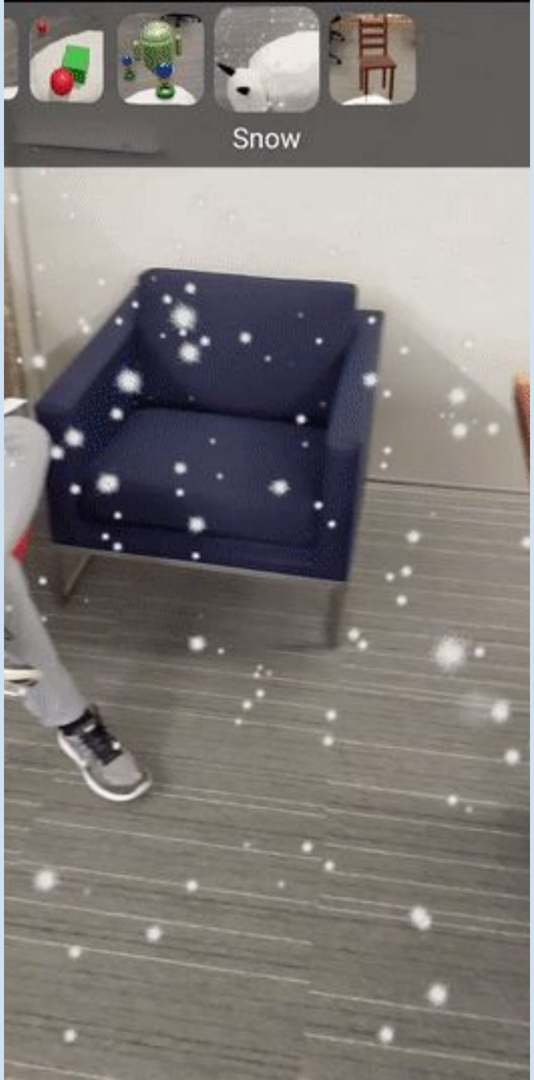


Localized Depth

Avatar Path Planning

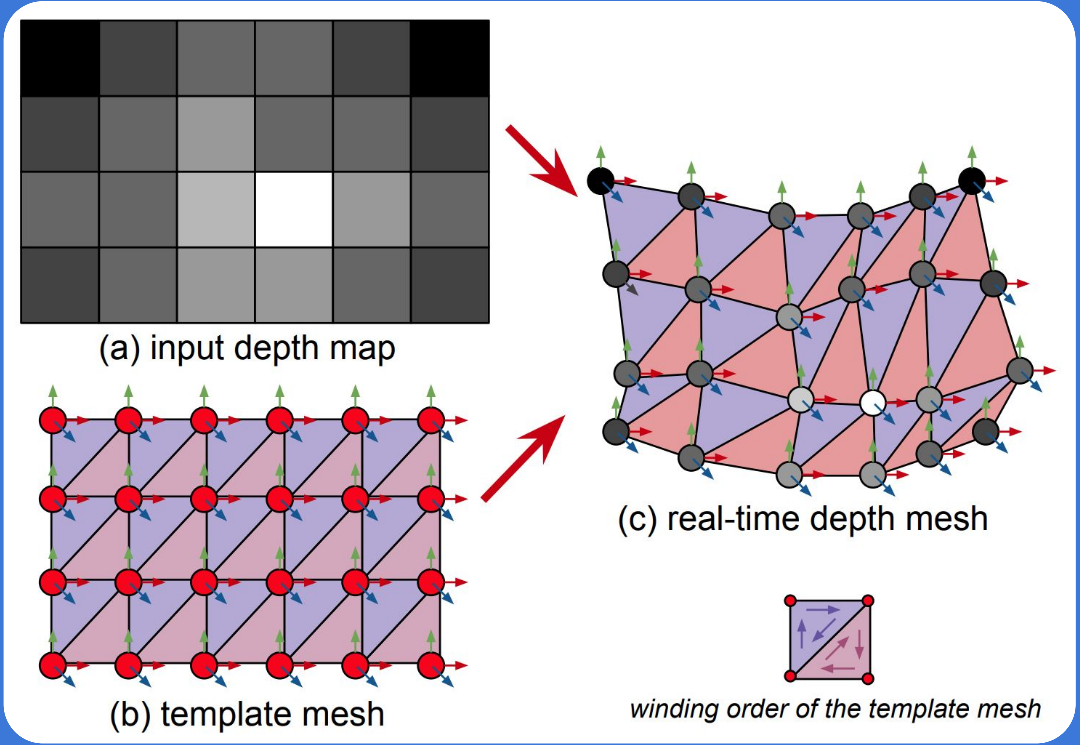
Localized Depth

Rain and Snow



Surface Depth

Use Cases



Surface Depth

Physics collider

Physics with depth mesh.

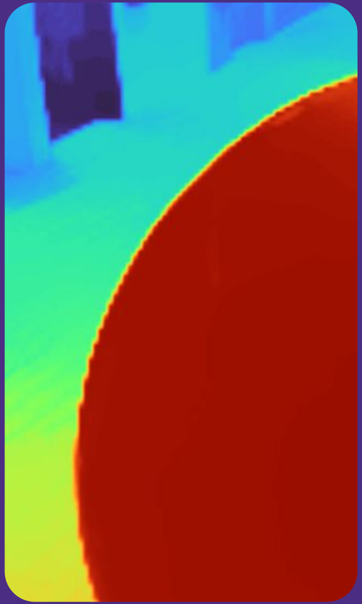
Texture decals with depth mesh.

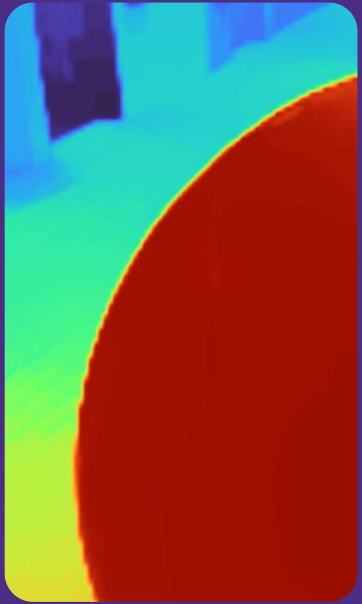
Surface Depth
3D Photo

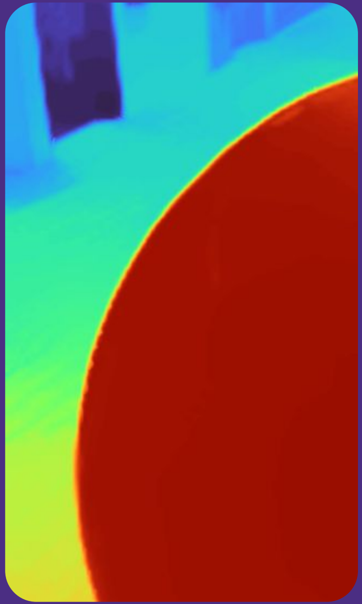
Projection mapping with depth mesh.

Dense Depth

Depth Texture - Antialiasing

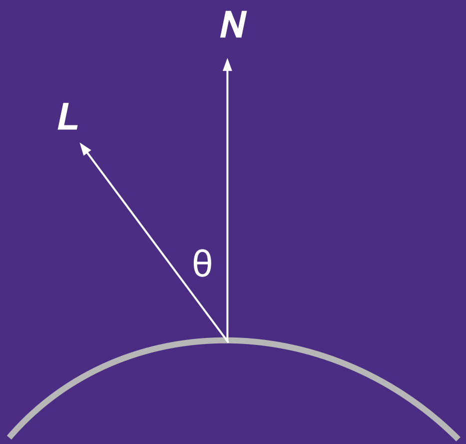






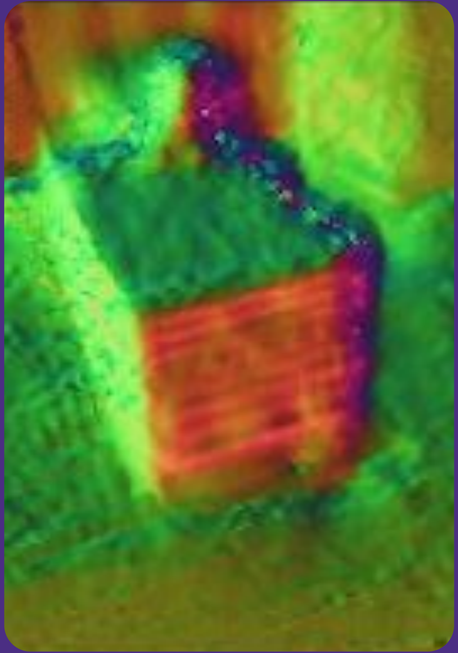
Dense Depth

Real-time relighting



Dense Depth

Why normal map does not work?



Dense Depth

Real-time relighting

Algorithm 3: Ray-marching-based Real-time Relighting.

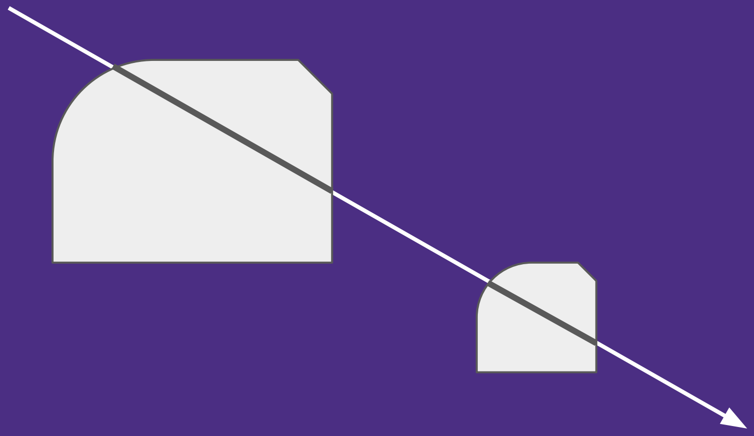
Input : Depth map $\mathbf{D}$, the camera image $\mathbf{I}$, camera intrinsic matrix $\mathbf{K}$, L light sources $\mathbb{L} = \{\mathscr{L}^i, i \in L\}$ with each light's location $\mathbf{v}_{\mathscr{L}}$ and intensity in RGB channels $\phi_{\mathscr{L}}$.

Output : Relighted image $\mathbf{O}$.

```
for each image pixel p ∈ depth map D in parallel do
    Sample p's depth value d ← D(p).
    Compute the corresponding 3D vertex v_p of the screen
      point p using the camera intrinsic matrix v_p with K:
      v_p = D(p) · K^{-1} [p, 1]
    Initialize relighting coefficients of v_p in RGB: φ_p ← 0.
    for each light 𝓛 ∈ light sources 𝕃 do
        Set the current photon coordinates v_o ← v_p.
        Set the current photon energy E_o ← 1.
        while v_o ≠ v_𝓛 do
            Compute the weighted distance between the
              photon to the physical environment
              Δd ← α|v_o^{xy} − v_𝓛^{xy}| + (1 − α)|v_o^z − v_𝓛^z|, α = 0.5.
            Decay the photon energy: E_o ← 95%E_o
            Accumulate the relighting coefficients:
              φ_p ← φ_p + ΔdE_oφ_𝓛.
            March the photon towards the light source:
              v_o ← v_o + (v_𝓛 − v_o)/S, here S = 10, depending
              on the mobile computing budget.
        end
    end
    Sample pixel's original color: Φ_p ← I(p).
    Apply relighting effect:
      O(p) ← γ · |0.5 − φ_p| · Φ_p^{1.5−φ_p} − Φ_p, here γ ← 3.
end
```

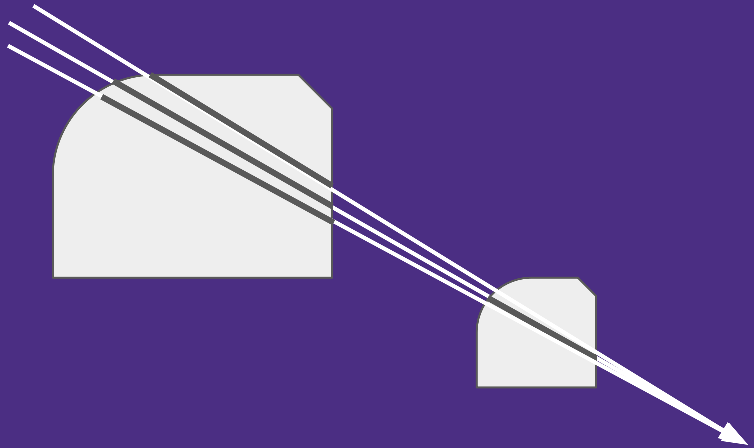
Dense Depth

Real-time relighting



Dense Depth

Real-time relighting



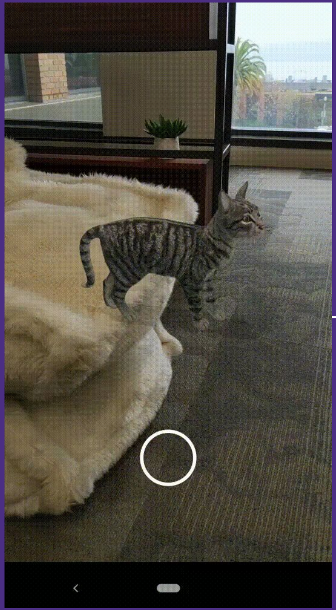
go/realtime-relighting, go/relit

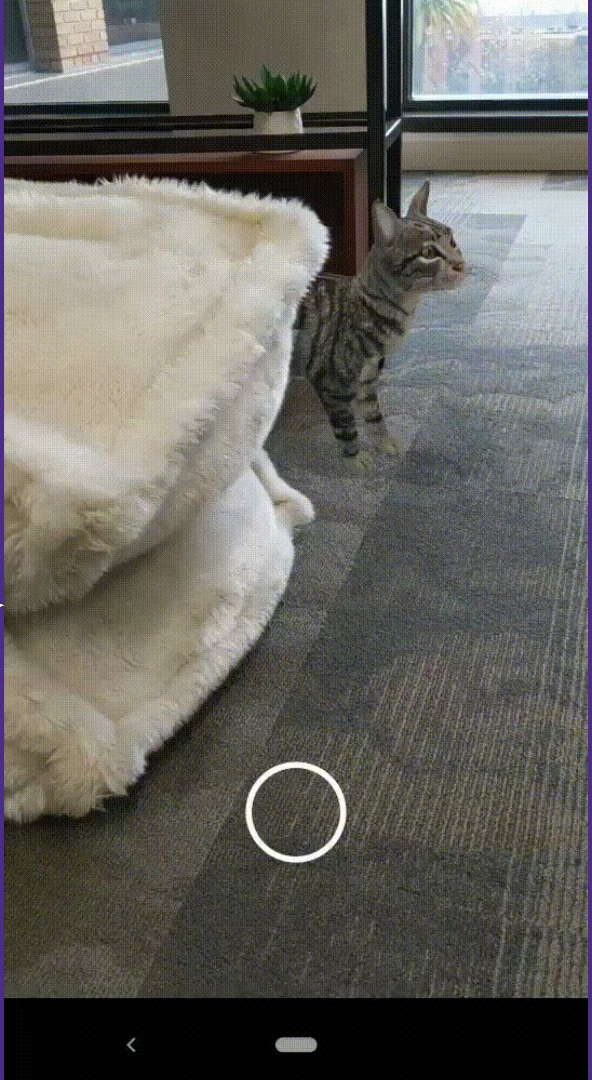
Dense Depth

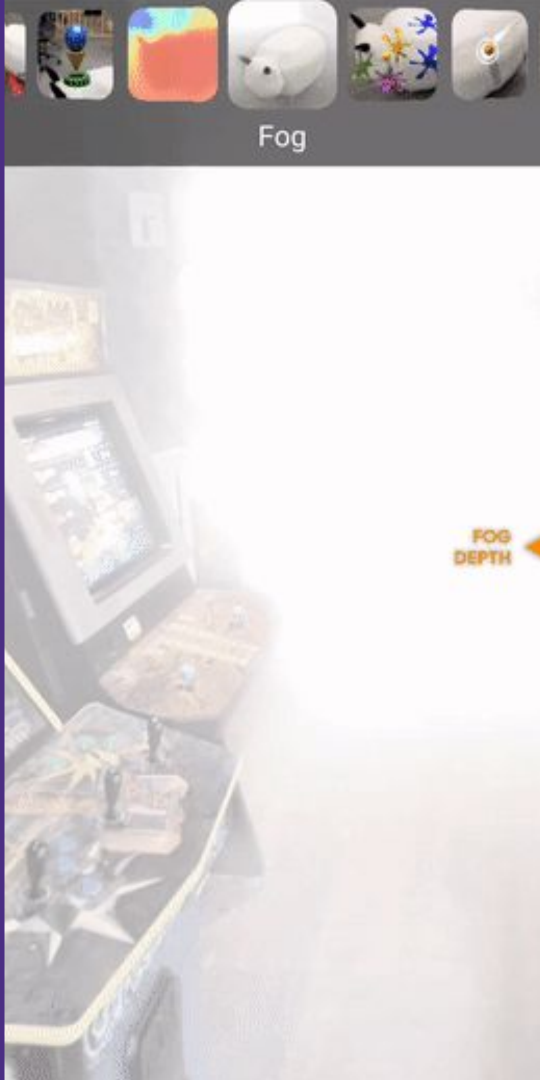
Wide-aperture effect

Dense Depth

Occlusion-based rendering

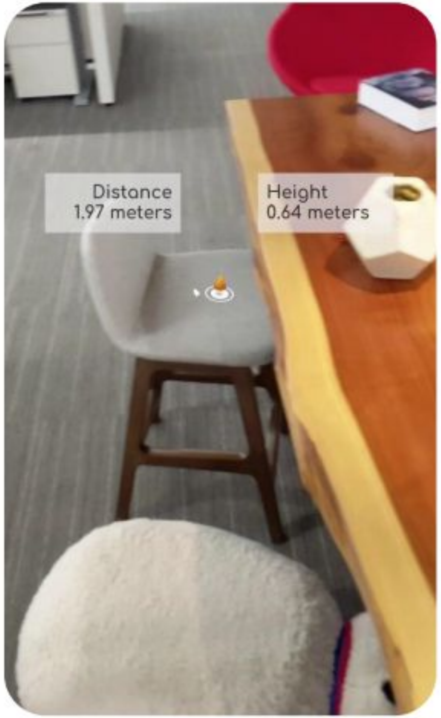


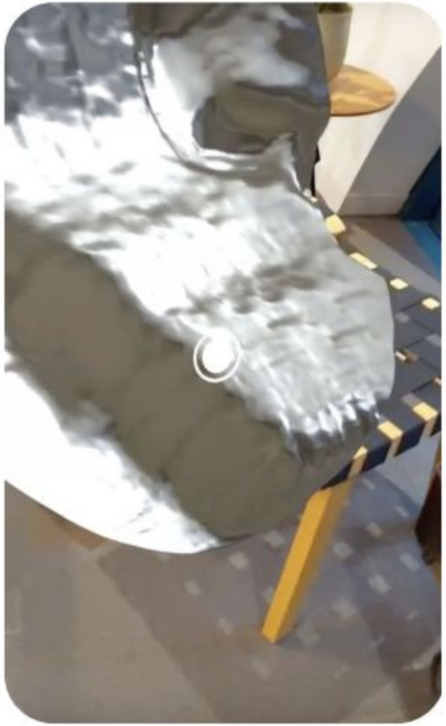


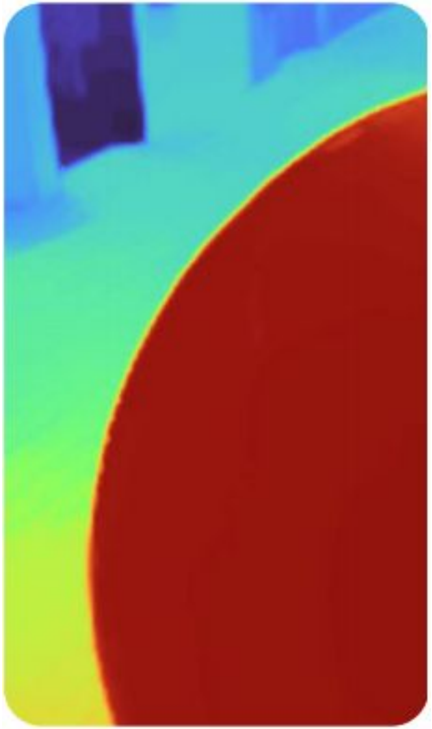


Experiments

DepthLab minimum viable application







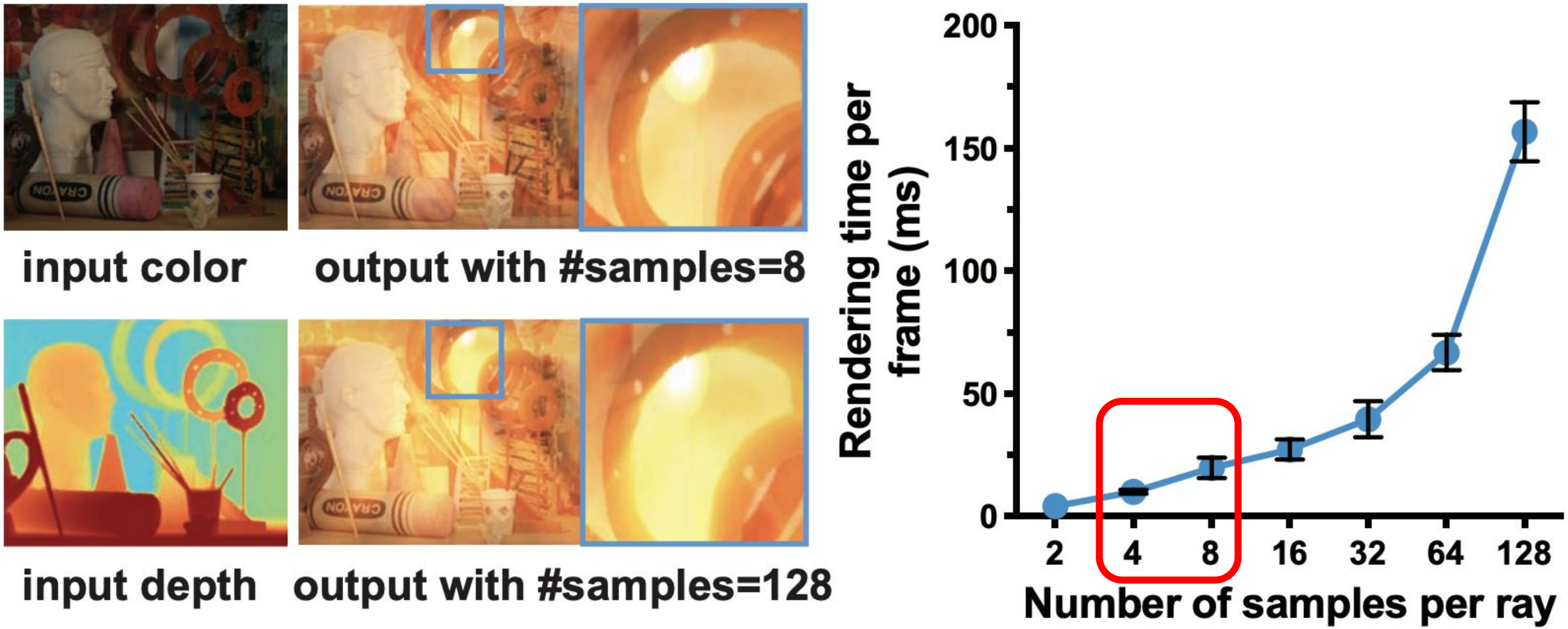
Experiments

General Profiling of MVP

Procedure	Timings (ms)
DepthLab's overall processing and rendering in Unity	8.32
DepthLab's data structure update and GPU uploading	1.63
Point Depth: normal estimation algorithm	< 0.01
Surface Depth: depth mesh update algorithm	2.41
Per-pixel Depth: visualization with single texture fetch	0.32

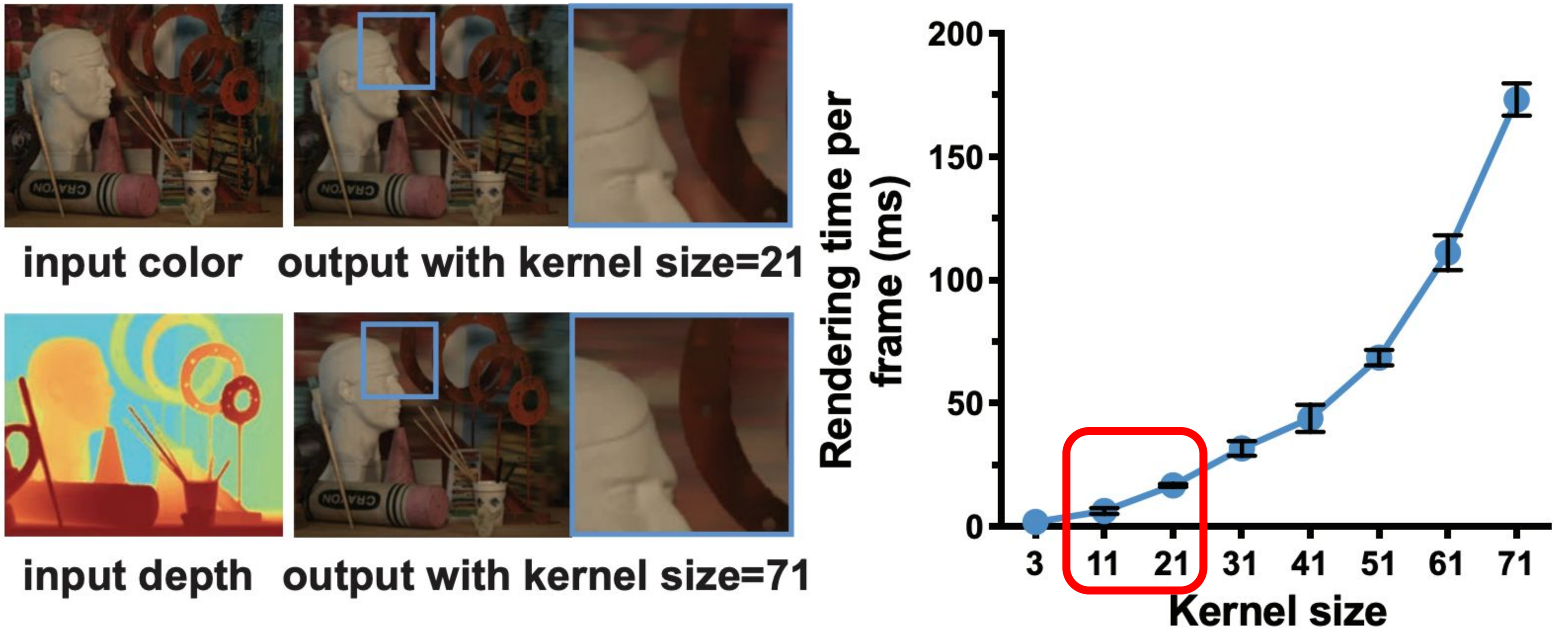
Experiments

Relighting



Experiments

Aperture effects



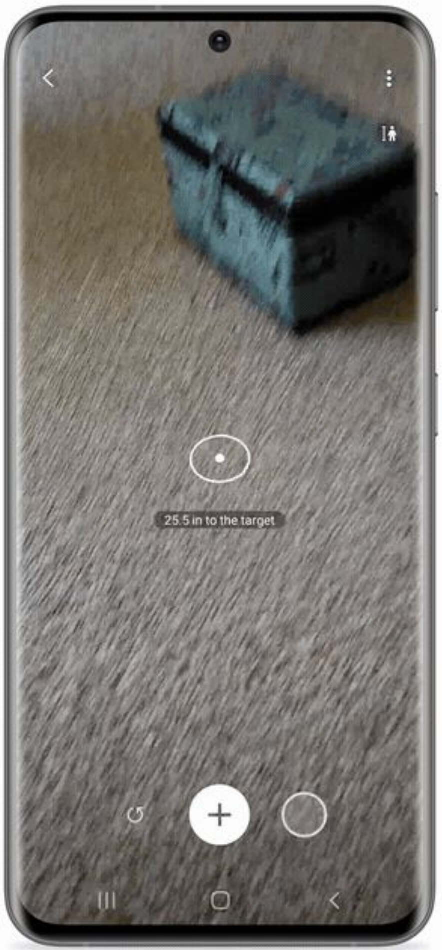
(a) examples of aperture effects (b) performance benchmark

Discussion

Deployment with partners

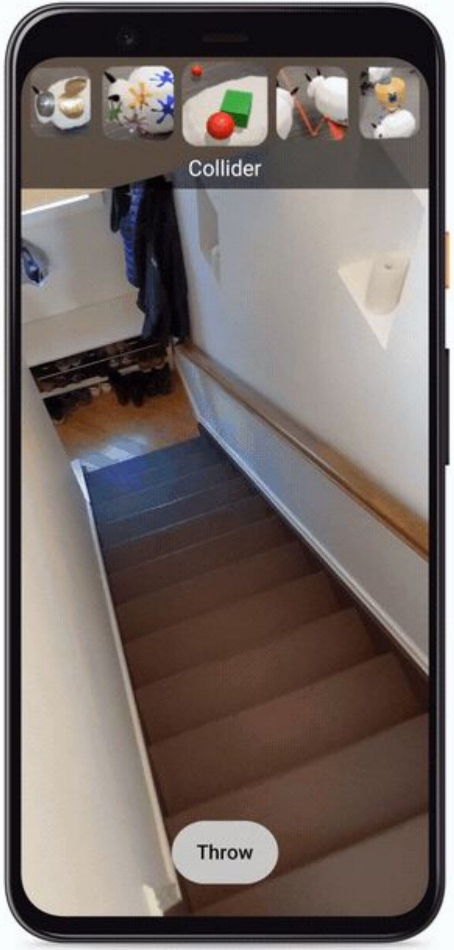
Discussion

Deployment with partners



Discussion

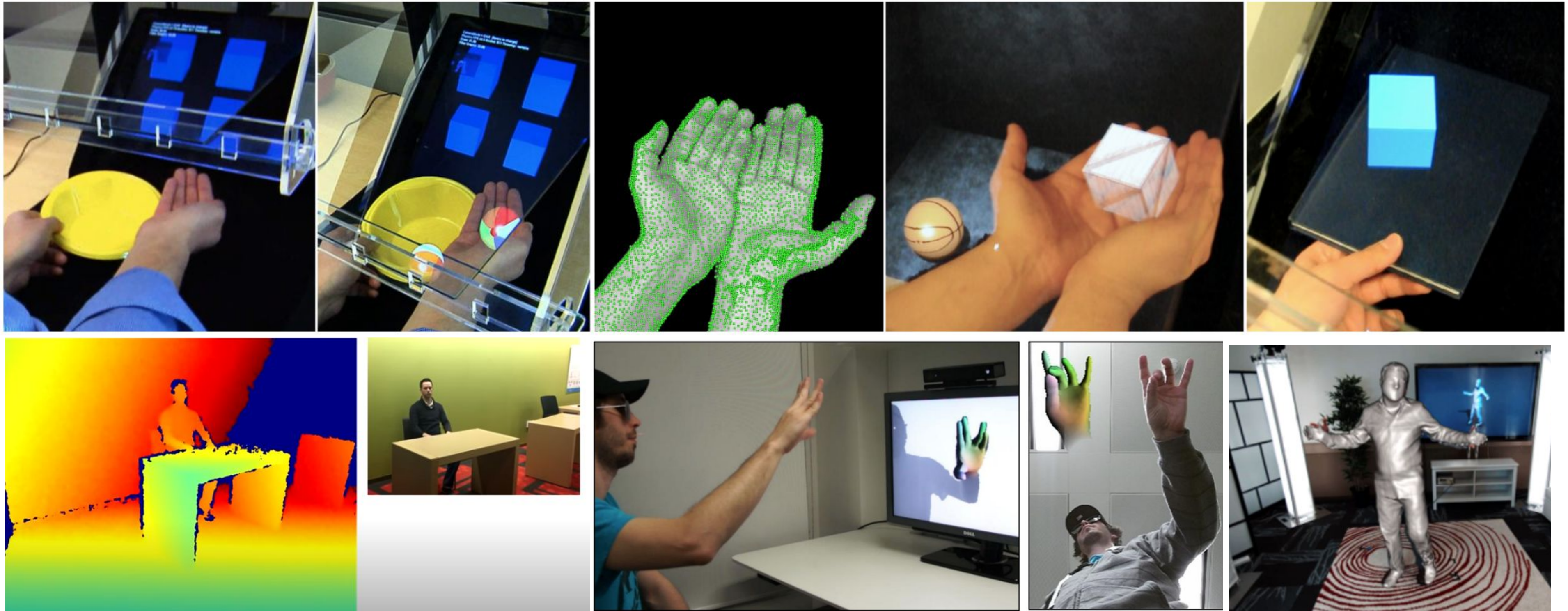
Deployment with partners



Limitations

Design space of dynamic depth

Dynamic Depth? HoloDesk, HyperDepth, Digits, Holoportation for mobile AR?

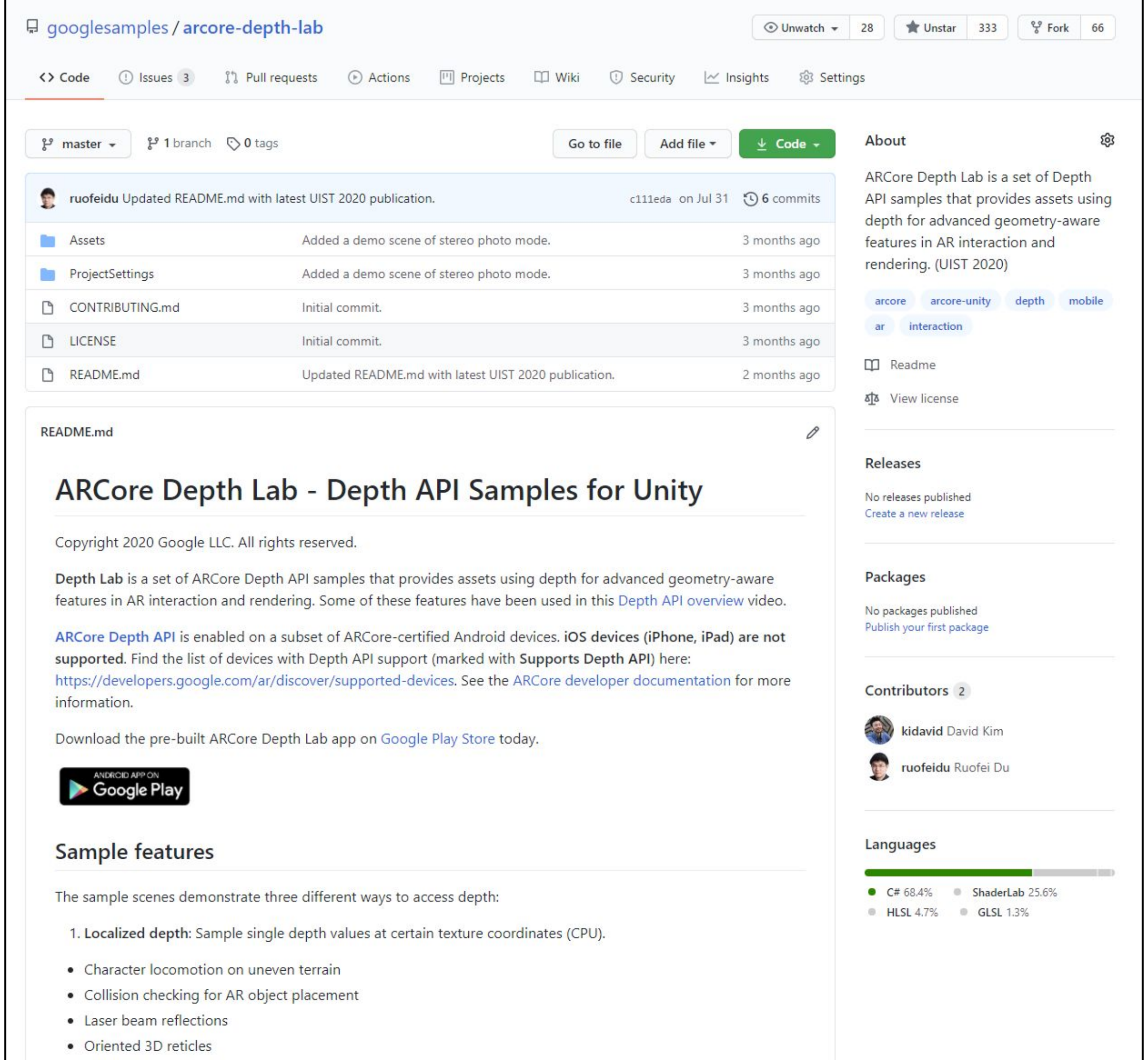


Envision

Design space of dynamic depth

GitHub

Please feel free to fork!



googlesamples / arcore-depth-lab

Unwatch 28 | Unstar 333 | Fork 66

Code | Issues 3 | Pull requests | Actions | Projects | Wiki | Security | Insights | Settings

master | 1 branch | 0 tags | Go to file | Add file | Code

ruofeidu Updated README.md with latest UIST 2020 publication. c111eda on Jul 31 | 6 commits

Assets	Added a demo scene of stereo photo mode.	3 months ago
ProjectSettings	Added a demo scene of stereo photo mode.	3 months ago
CONTRIBUTING.md	Initial commit.	3 months ago
LICENSE	Initial commit.	3 months ago
README.md	Updated README.md with latest UIST 2020 publication.	2 months ago

README.md

ARCore Depth Lab - Depth API Samples for Unity

Copyright 2020 Google LLC. All rights reserved.

Depth Lab is a set of ARCore Depth API samples that provides assets using depth for advanced geometry-aware features in AR interaction and rendering. Some of these features have been used in this Depth API overview video.

ARCore Depth API is enabled on a subset of ARCore-certified Android devices. **iOS devices (iPhone, iPad) are not supported**. Find the list of devices with Depth API support (marked with **Supports Depth API**) here: https://developers.google.com/ar/discover/supported-devices. See the ARCore developer documentation for more information.

Download the pre-built ARCore Depth Lab app on Google Play Store today.

ANDROID APP ON Google Play

Sample features

The sample scenes demonstrate three different ways to access depth:

1. **Localized depth**: Sample single depth values at certain texture coordinates (CPU).

- Character locomotion on uneven terrain
- Collision checking for AR object placement
- Laser beam reflections
- Oriented 3D reticles

About

ARCore Depth Lab is a set of Depth API samples that provides assets using depth for advanced geometry-aware features in AR interaction and rendering. (UIST 2020)

arcore | arcore-unity | depth | mobile | ar | interaction

Readme

View license

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Contributors 2

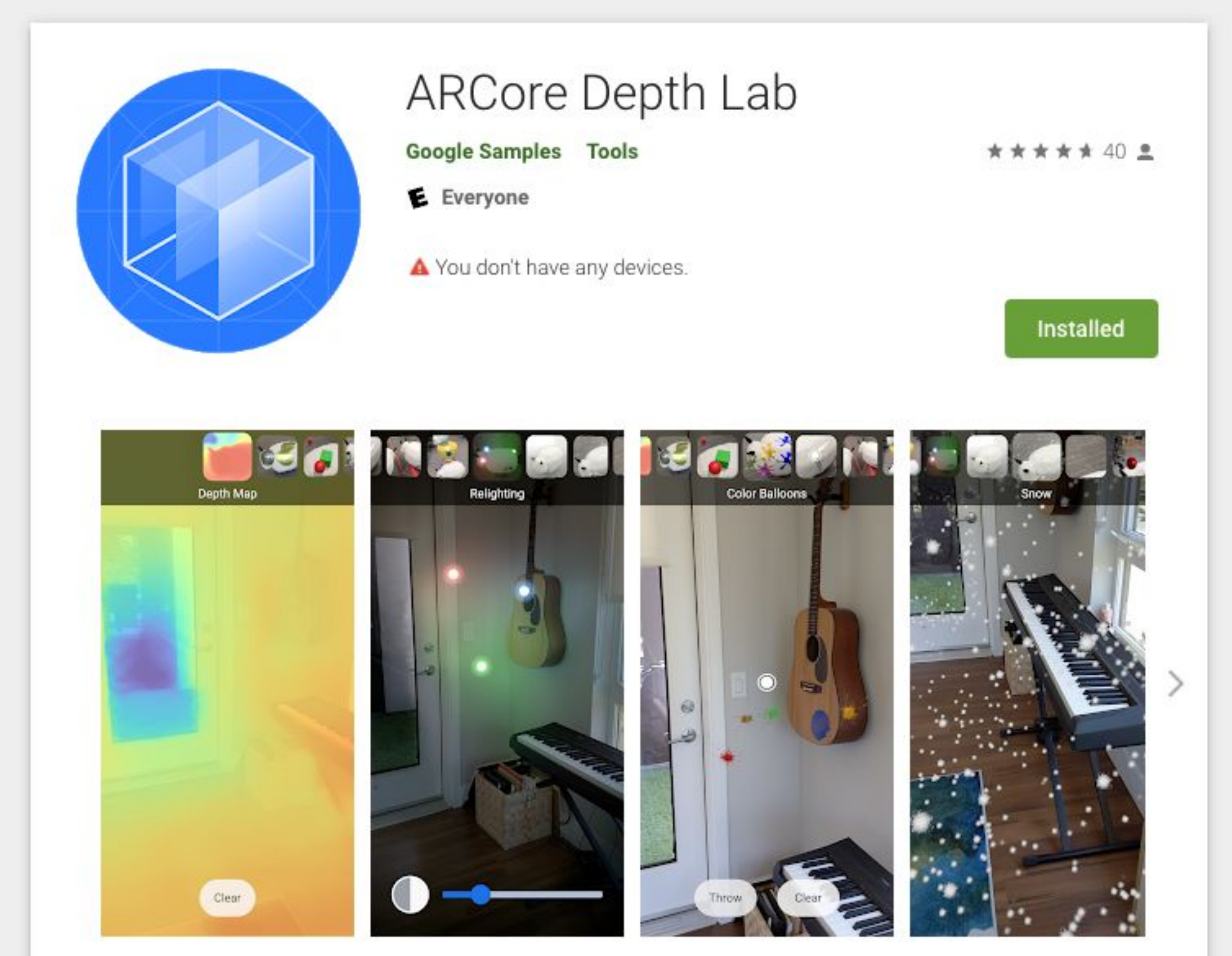
kidavid David Kim
ruofeidu Ruofei Du

Languages

C# 68.4% | ShaderLab 25.6% | HLSL 4.7% | GLSL 1.3%

Play Store

Try it yourself!



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

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 | ACM UIST 2020

Thank you!

DepthLab | UIST 2020

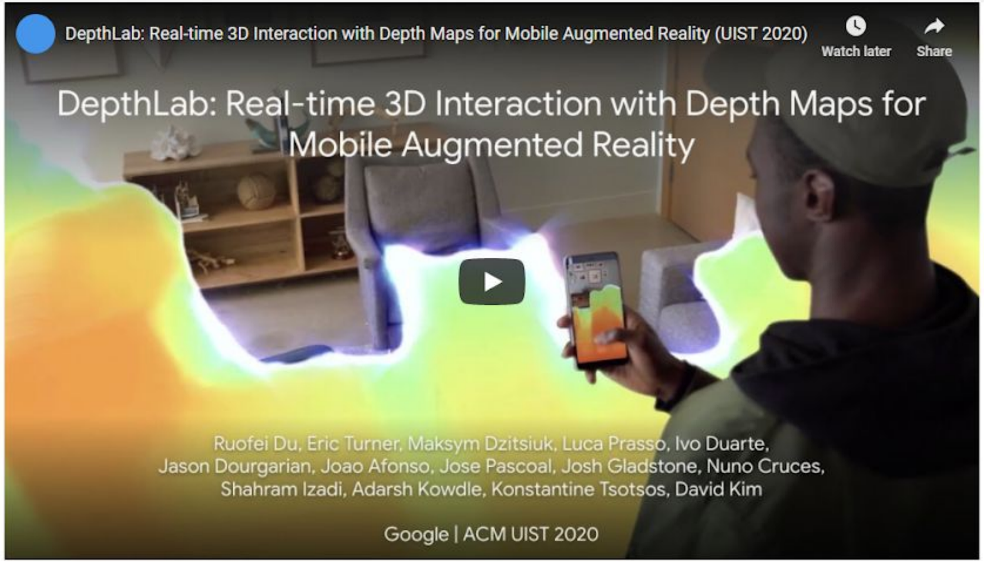
DEPTHLAB: REAL-TIME 3D INTERACTION WITH DEPTH MAPS FOR MOBILE AUGMENTED REALITY

ACM UIST 2020

Download PDF (6 MB) or Low-Res PDF (4 MB)

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



Demo

DepthLab | UIST 2020

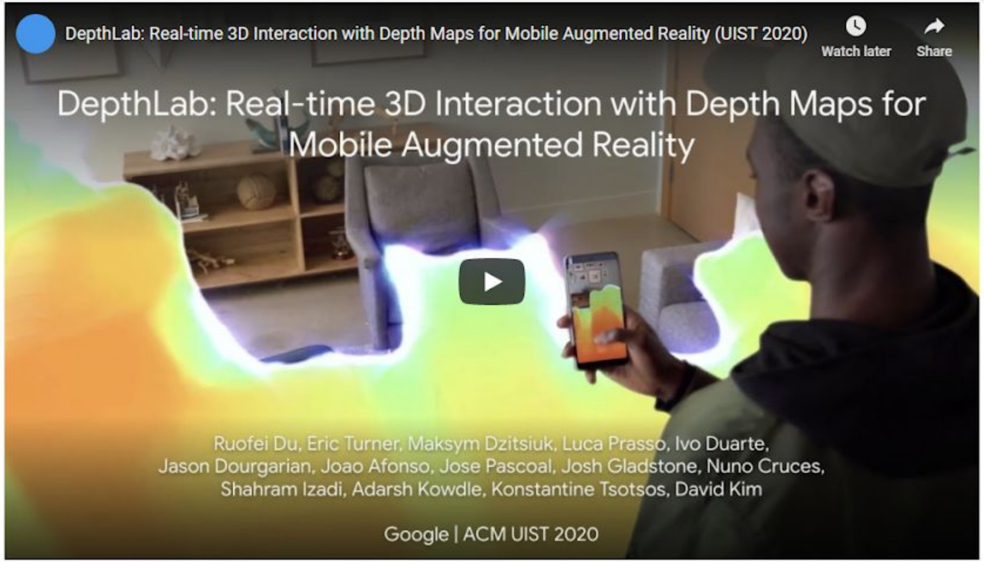
DEPTHLAB: REAL-TIME 3D INTERACTION WITH DEPTH MAPS FOR MOBILE AUGMENTED REALITY

ACM UIST 2020

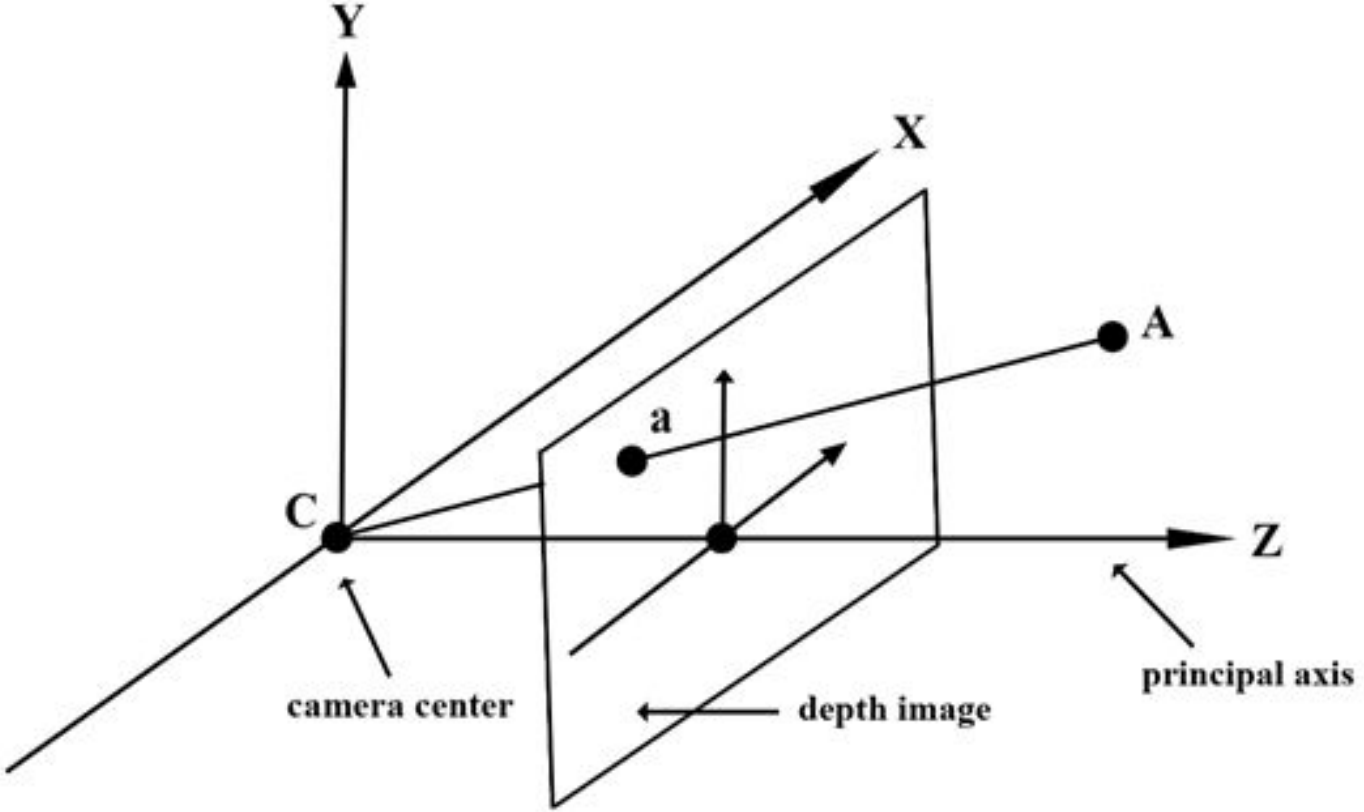
Download PDF (6 MB) or Low-Res PDF (4 MB)

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



Introduction
Depth Map



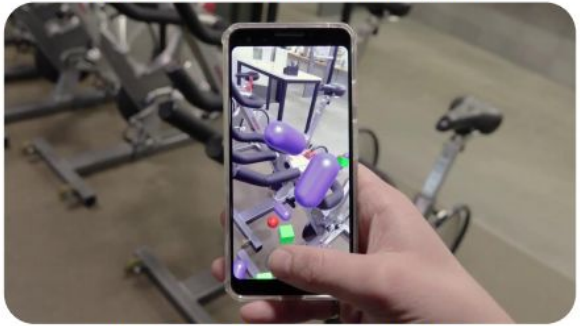
Introduction

Depth Lab

(a) oriented reticles and splats

(b) ray-marching-based scene relighting

(c) depth visualization and particles



(d) geometry-aware collisions

(e) 3D-anchored focus and aperture effect

(f) occlusion and path planning

Thank you!

www.duruofei.com

DepthLab: Real-Time 3D Interaction With Depth Maps for Mobile Augmented Reality

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, and David Kim

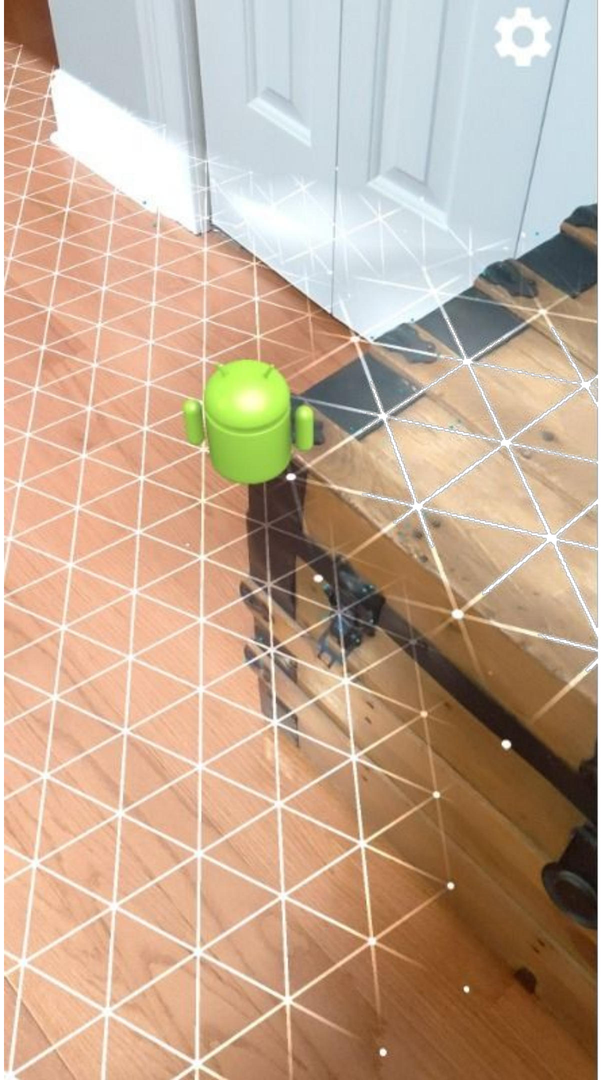
Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology (UIST), 2020.

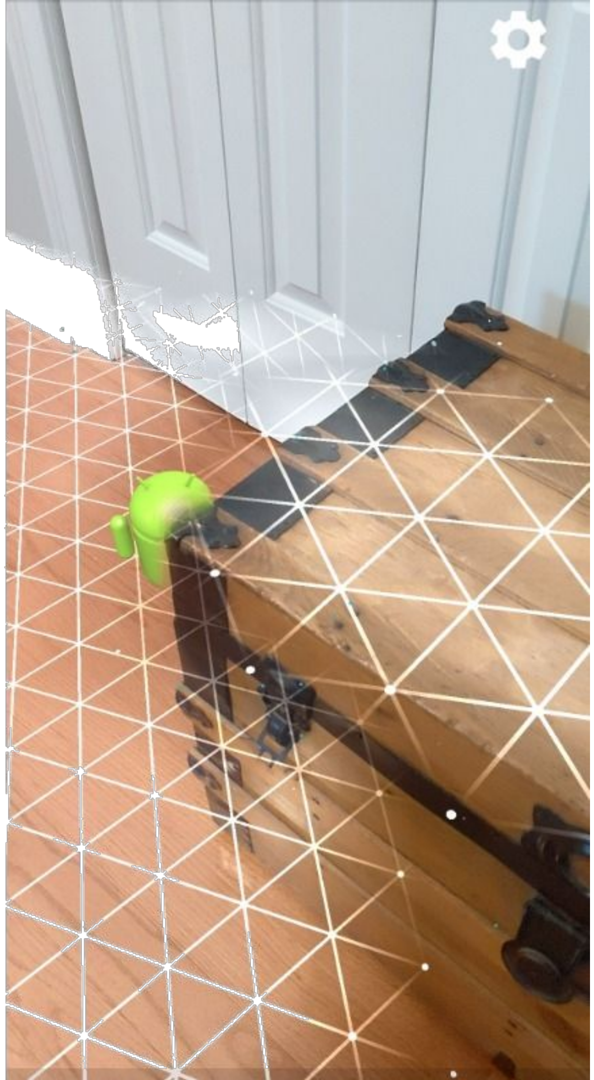
pdf, lowres | website, code, demo, supp | cite

Occlusion is a critical component for AR realism!
Correct occlusion helps ground content in reality, and makes virtual objects feel as if they are actually in your space.

Introduction

Motivation





Depth Mesh

Generation

Algorithm 2: Real-time Depth Mesh Generation.

Input : Depth map $\mathbf{D}$, its dimension $w \times h$, and depth discontinuity threshold $\Delta d_{\max} = 0.5$.

Output : Lists of mesh vertices $\mathbb{V}$ and indices $\mathbb{I}$.

In the initialization stage on the CPU:

for $x \in [0, w-1]$ **do**
  for $y \in [0, h-1]$ **do**
    Set the pivot index: $I_0 \leftarrow y \cdot w + x$.
    Set the neighboring indices:
    $I_1 \leftarrow I_0 + 1,\ I_2 \leftarrow I_0 + w,\ I_3 \leftarrow I_2 + 1.$
    Add the vertex $(x/w, y/h, 0)$ to $\mathbb{V}$.
  end
end

In the rendering stage on the CPU or GPU:

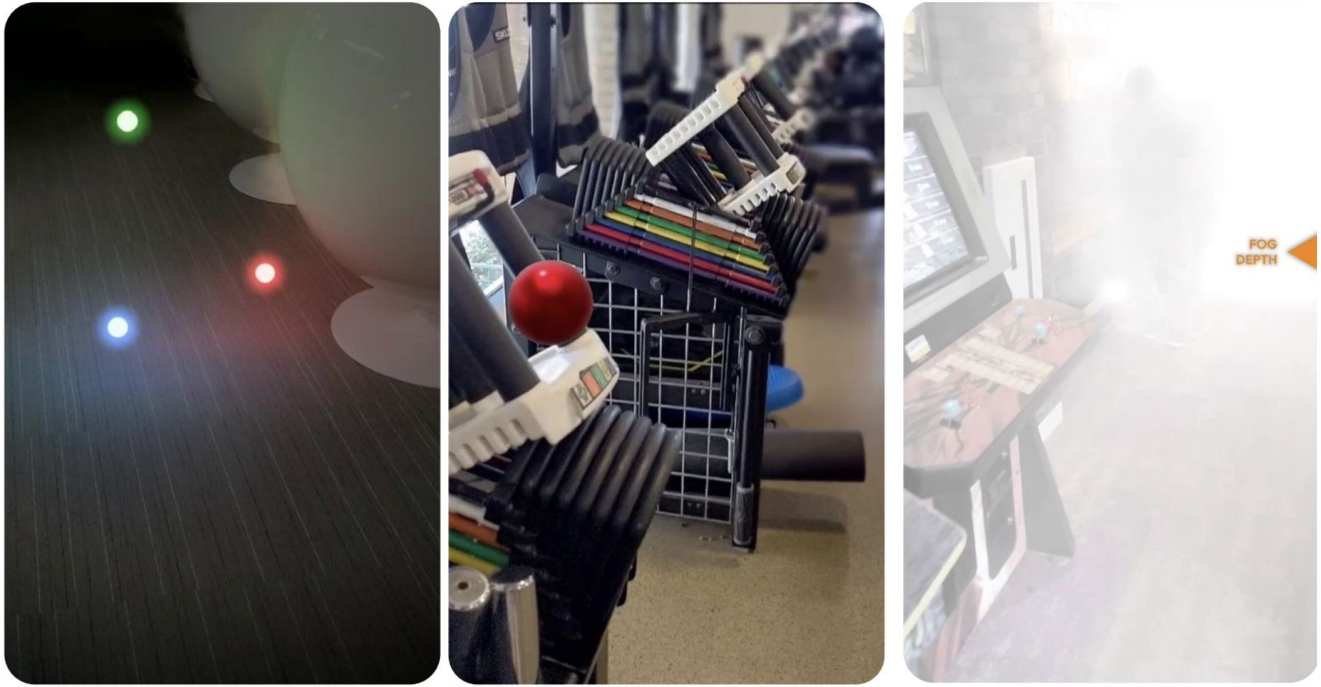
for *each vertex* $v \in \mathbb{V}$ **do**
  Look up v's corresponding screen point $\mathbf{p}$.
  Fetch v's depth value $d_0 \leftarrow \mathbf{D}(\mathbf{p})$.
  Fetch v's neighborhoods' depth values:
  $d_1 \leftarrow \mathbf{D}(\mathbf{p} + (1,0)),\ d_2 \leftarrow \mathbf{D}(\mathbf{p} + (0,1)),$
  $d_3 \leftarrow \mathbf{D}(\mathbf{p} + (1,1)).$
  Compute average depth $\bar{d} \leftarrow \frac{d_0+d_1+d_2+d_3}{4}$.
  Let $\mathbf{d} \leftarrow [d_0, d_1, d_2, d_3]$.
  if $\textit{any}\left(\textit{step}\left(\Delta d_{\max}, |\mathbf{d} - \bar{d}|\right)\right) = 1$ **then**
    Discard v due to large depth discontinuity.
  end
end

Localized Depth

Avatar Path Planning

Dense Depth

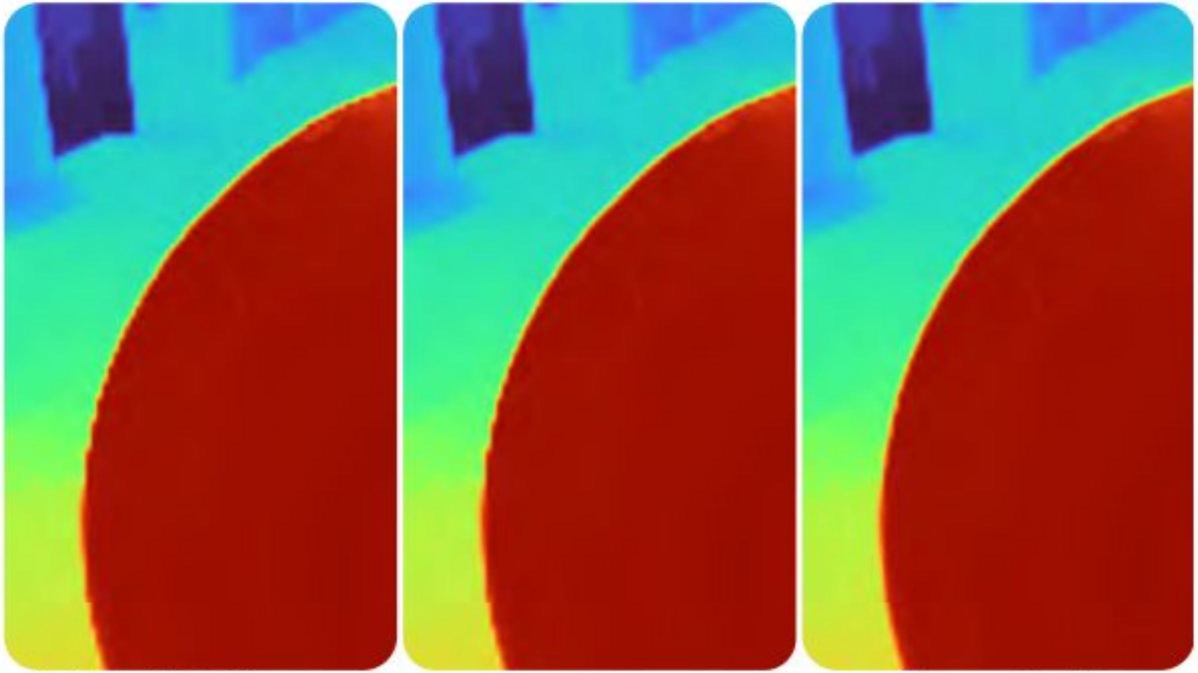
Depth Texture



(a) relighting effect (b) aperture effect (c) fog effect

Introduction

Depth Map



(a) input depth map (bilinearly filtered)

(b) result depth map with FXAA

(c) result with depth-guided FXAA

Taxonomy

Depth Usage

Depth-based Interaction Design Space

Geometry-aware ...	Depth Interaction ...	Visual Effects of ...
Rendering	**Interface**	**Static**
occlusion shadows relighting	3D cursor bounding-box region selection ...	Bokeh effect triplanar mapping aligned AR text ...
Actions	**Gestures**	**Dynamic**
physics path planning free-space check ...	static hand dynamic motion 3D touch ...	depth transition light painting surface ripples ...

Introduction
Depth Map

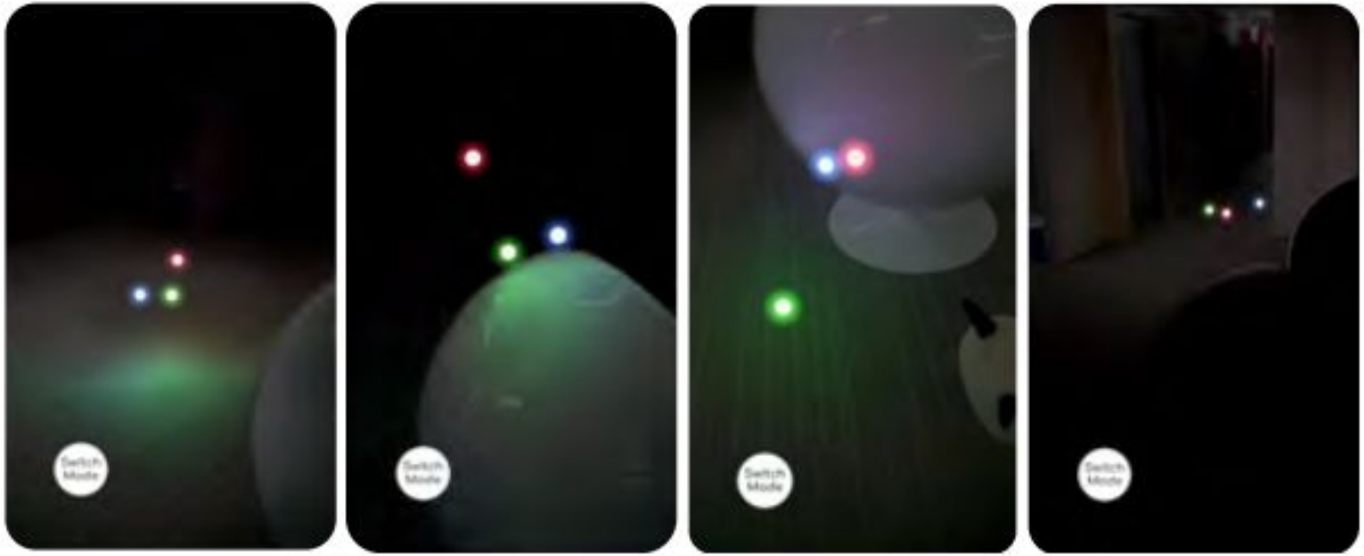


Figure 12. Given a dense depth texture, a camera image, and virtual light sources, we altered the lighting of the physical environment by tracing occlusions along the light rays in real time.

Introduction

Depth Map

Figure 13. Wide-aperture effect focused on a world-anchored point on a flower from different perspectives. Unlike traditional photography software, which only anchors the focal plane to a screen point, DepthLab allows users to anchor the focal point to a physical object and keep the object in focus from even when the viewpoint changes. Please zoom in to compare the focus and out-of-focus regions.