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

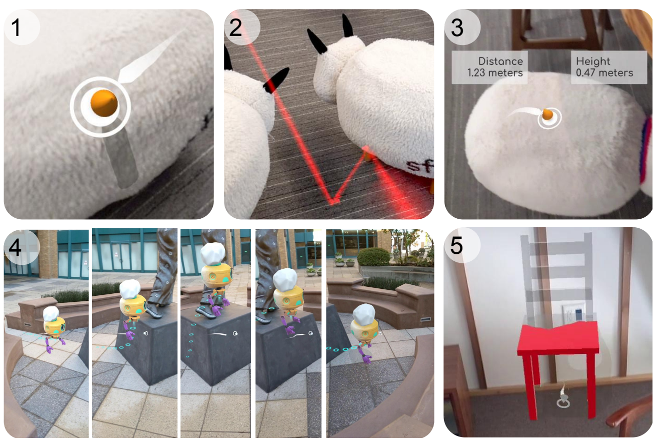

Figure 1. Implementation examples of geometry-aware AR features 1–5 with localized depth use cases. Please refer to the supplementary video for live demonstration.

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.

2. **Laser reflection:** Render a virtual laser from the user to physical objects along the camera's principle axis by touching the screen. The laser should be reflected when reaching a surface. The hit and reflection algorithms should be reusable for mobile AR developers.
Implemented in DepthLab: Yes.

3. **Physical measurement:** Measure the distance and height of an arbitrary physical point in meters by touching a pixel on the phone screen.
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

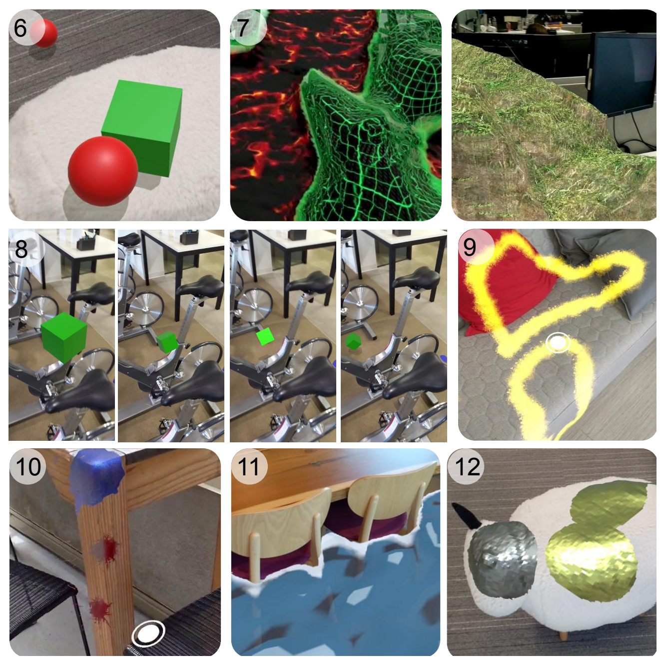

Figure 2. Implementation examples of geometry-aware AR features 6–9.

6. **Virtual shadows:** Render geometry-aware shadows [11] that are cast onto physical surfaces. The shadow may be integrated with any mobile AR application with virtual objects.
Implemented in DepthLab: Yes.

7. **Environmental texturing:** Re-texture physical surfaces with other materials, e.g. lava, grids, grass. This technique could also be used to replace the ceiling with the star map of your location or generate a terrain with grass, vegetation, or rock.
Implemented in DepthLab: Yes.

†Corresponding author. Please contact Ruofei Du at me@duruofei.com and/or David Kim at contact@davidkim.de.

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.

10. **AR paintballs:** Allow the user to throw color balloons onto physical objects. The balloons should explode as texture decals onto the surfaces they hit.
Implemented in DepthLab: Yes.

11. **AR flooding:** Detect empty ground regions and render water-flooding effects in the physical environment. The water mesh is procedurally generated where the environment's elevation is lower than the predefined water level.
Implemented in DepthLab: Yes.

12. **Mesh freezing:** Allow the user to freeze a portion of the screen-space mesh, change its material, and observe it from another perspective.
Implemented in DepthLab: Yes.

13. **Object-triggered geometry-aligned tags:** Anchor labels on top of the recognized object by using object recognition models, operating as a virtual label printer.
Implemented in DepthLab: No.
Could be implemented by looking for the highest surface of the object and attaching virtual tags to it. However, this method would be best implemented with semantic segmentation algorithms.

14. **Perspective illusion art:** Capture an image of the environment from a single point of view, then decompose the image into a 3D pattern when the user shifts the viewpoint. Project a texture on the depth map and keep the original 6-DoF pose of the projection.
Implemented in DepthLab: No.

Depth Representation Requirement: Dense Depth

15. **Object occlusion:** Occlude virtual objects placed behind physical objects. This component is useful for almost all mobile AR application with virtual objects.
Implemented in DepthLab: Yes.

16. **3D-anchored focus and aperture effect:** Render "depth-of-field" effects that simulate a DSLR camera. The user may anchor the focus point to a physical object and set the focal plane. The pixels that are outside the simulated depth of field are blurred out.
Implemented in DepthLab: Yes.

17. **Relighting effects:** Relight the physical environment with virtual light sources. The user may adjust the virtual light intensity, color, and position.
Implemented in DepthLab: Yes.

18. **Snow effects:** Generate snow particles randomly outside the screenspace and make them fall to the ground with random velocity. Each particle vanishes when it lands on a surface.
Implemented in DepthLab: Yes.

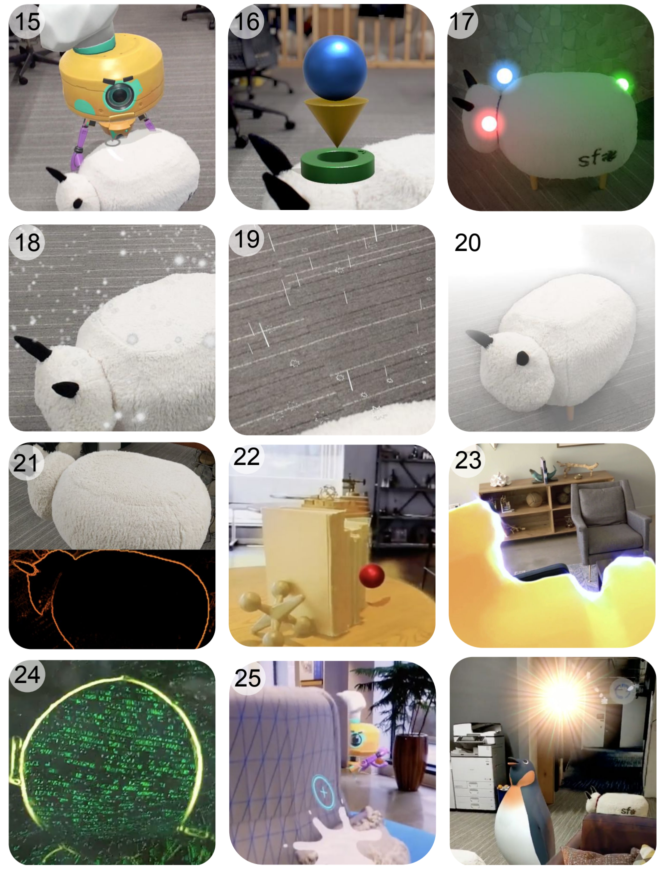

Figure 3. Implementation examples of geometry-aware AR features 15–25 with dense depth.

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 from close to far, or far to close.
Implemented in DepthLab: Yes.

24. **"The Matrix" effect:** Embed animated ASCII code into the physical environment for AR gaming purposes.
Implemented in DepthLab: Yes.

25. **Design a "hide and seek" game:** Spawn virtual avatars, occluded behind physical obstacles. The user may look around and tap on the avatar on the phone screen to catch them.
Implemented in DepthLab: Yes.

26. **Render wigglegram and kinetic-depth images (3D photos) [3]**: Aid the visualization of the three-dimensional structure of a scene by leveraging the motion of the mobile device in the rendering.
Implemented in DepthLab: Yes.

27. **Remove objects with depth-based image in-painting:** Dense depth map may assist image-based Poisson blending or deep-learning techniques for object removal.
Implemented in DepthLab: No.

28. **Compress video for teleconferencing with depth data:** After segmenting out the background with the dense depth map, the application may only transmit the foreground pixels for video conferencing.
Implemented in DepthLab: No.

Depth Representation Requirement: Persistent Voxels

All ideas with dynamic voxels are not supported by DepthLab so far.

29. **Scan commodity objects or humans as 3D models [7]**: The 3D model may be further used for online shopping, virtual design, and entertainment industries. The user would be required to take photographs from every perspective of the object.

30. **Segment physical objects with user-guided strokes [9]**: This method requires the system to keep track of the strokes and currently segmented portion of the mesh.

31. **Music visualization**: Visualize music by animating the point cloud of the physical world[1].

32. **Semantic object labelling**: Label physical objects with semantic classes [6] and colorize each object based on its corresponding label or overlay text next to the object.

33. **Virtual mirrors**: Render virtual mirrors with photo-realistic reflections [10]. The system must memorize persistent meshes around the user.

34. **Generate occlusion-aware spatial sound effects**: Leverage ambient sound propagation techniques [13] to simulate the spatial sound with persistent reconstructed meshes.

Depth Requirement: Dynamic Voxels

All ideas with dynamic voxels are not implemented in DepthLab so far, but may be reproduced on mobile phones with time-of-flight sensors.

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.

[1]Example concept of music visualization with voxels in VR: https://www.shadertoy.com/view/wsSXzh

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 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, 14.

[6] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. 2017. Mask R-CNN. In *Proceedings of the IEEE international conference on computer vision*. 2961–2969.

[7] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, and Andrew Fitzgibbon. 2011. KinectFusion: Real-Time 3D Reconstruction and Interaction Using a Moving Depth Camera. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST '11)*. ACM, Santa Barbara, California, USA, 559–568. DOI: http://dx.doi.org/10.1145/2047196.2047270

[8] Jonathan Taylor, Lucas Bordeaux, Thomas Cashman, Bob Corish, Cem Keskin, Toby Sharp, Eduardo Soto, David Sweeney, Julien Valentin, Benjamin Luff, Arran Topalian, Erroll Wood, Sameh Khamis, Pushmeet Kohli, Shahram Izadi, Richard Banks, Andrew Fitzgibbon, and Jamie Shotton. 2016. Efficient and Precise Interactive Hand Tracking Through Joint, Continuous Optimization of Pose and Correspondences. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 1–12. DOI: http://dx.doi.org/10.1145/2897824.2925965

[9] Julien Valentin, Vibhav Vineet, Ming-Ming Cheng, David Kim, Jamie Shotton, Pushmeet Kohli, Matthias Nießner, Antonio Criminisi, Shahram Izadi, and Philip Torr. 2015. Semanticpaint: Interactive 3d Labeling and Learning at Your Fingertips. *ACM Transactions on Graphics (TOG)* 34, 5 (2015), 154. DOI: http://dx.doi.org/10.1145/2751556

[10] Thomas Whelan, Michael Goesele, Steven J Lovegrove, Julian Straub, Simon Green, Richard Szeliski, Steven Butterfield, Shobhit Verma, and Richard A Newcombe. 2018. Reconstructing Scenes with Mirror and Glass Surfaces. *ACM Trans. Graph.* 37, 4 (2018), 102:1–11. DOI:http://dx.doi.org/10.1145/3197517.3201319

[11] Lance Williams. 1978. Casting Curved Shadows on Curved Surfaces. In *Proceedings of the 5th Annual Conference on Computer Graphics and Interactive Techniques*. 270–274. DOI: http://dx.doi.org/10.1145/965139.807402

[12] Robert Xiao, Julia Schwarz, Nick Throm, Andrew D Wilson, and Hrvoje Benko. 2018. MRTouch: Adding Touch Input to Head-Mounted Mixed Reality. *IEEE Transactions on Visualization and Computer Graphics* 24, 4 (2018), 1653–1660. DOI: http://dx.doi.org/10.1109/TVCG.2018.2794222

[13] Zechen Zhang, Nikunj Raghuvanshi, John Snyder, and Steve Marschner. 2018. Ambient Sound Propagation. *ACM Transactions on Graphics (TOG)* 37, 6 (2018), 1–10. DOI:http://dx.doi.org/10.1145/3272127.3275100