Real-time High Resolution 3D Data on the HoloLens

Mathieu Garon* Pierre-Olivier Boulet† Jean-Philippe Doiron‡ Luc Beaulieu§ Jean-François Lalonde¶
Laval University Frima Studio, Inc. Laval University

(a) User wearing our system (b) Scene (c) Scene observed through HoloLens

Figure 1: By combining a high resolution depth camera (Intel RealSense) with a Microsoft HoloLens headset (a), we can use object detection algorithms to precisely locate small objects (b) and overlay object-dependent information in the HoloLens field of view (c). Our resulting system is tetherless, and allows the use of algorithms that exploit high resolution depth information for HoloLens applications.

Abstract

The recent appearance of augmented reality headsets, such as the Microsoft HoloLens, is a marked move from traditional 2D screen to 3D hologram-like interfaces. Striving to be completely portable, these devices unfortunately suffer multiple limitations, such as the lack of real-time, high quality depth data, which severely restricts their use as research tools. To mitigate this restriction, we provide a simple method to augment a HoloLens headset with much higher resolution depth data. To do so, we calibrate an external depth sensor connected to a computer stick that communicates with the HoloLens headset in real-time. To show how this system could be useful to the research community, we present an implementation of small object detection on HoloLens device.

1 Introduction

As one of the first AR headsets available today on the market, the Microsoft HoloLens is bound to have a profound impact on the development of AR applications. For the first time, an intuitive and easy-to-use device is available to developers, who can use it to deploy AR applications to an ever expanding range of users and customers. What is more, that device is itself a portable computer, capable of operating without being connected to an external machine, making it well-suited for a variety of applications.

A downside of this portability is that access to the raw data provided by HoloLens sensors is not available. This severely restricts the use of the HoloLens as a research tool: by being forced to exclusively use the provided API functionality, future research using the HoloLens is quite limited indeed. Of note, the unavailability of high resolution depth data prohibits the development of novel object detection [7, 9], SLAM [5, 1], or tracking [6, 8] algorithms to name just a few, on-board these devices.

*email: mathieu.garon.2@ulaval.ca
†email: pierre-olivier.boulet.1@ulaval.ca
‡email: jean-philippe.doiron@frimastudio.com
§email: luc.beaulieu@frimastudio.com
¶email: jflalonde@gel.ulaval.ca

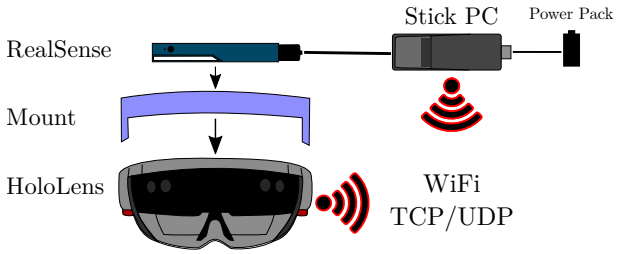

Figure 2: Overview of hardware setup. We attach an Intel RealSense RGBD camera on a HoloLens unit via a custom mount. The RealSense is connected to a stick PC via USB. This PC can then relay the high resolution depth data—or information such as detected objects computed from it—back to the HoloLens via WiFi.

In this poster, we present a system that bypasses this limitation and provides high resolution 3D data to HoloLens applications in real-time. The 3D data is accurately registered to the HoloLens reference frame and allows the integration of any depth- or 3D-based algorithm for use on the HoloLens. As seen in fig. 1, our key idea is to couple a depth camera (an Intel Realsense in our case—but other such portable RGBD cameras could be used as well) with a HoloLens unit using a custom-made attachment, and to transfer the depth information in real-time via a WiFi connection operated on a stick PC, also attached to the HoloLens. We do so without sacrificing the portability of the device: our system is still tetherless, as is the original HoloLens.

The remainder of this short paper will describe the hardware setup in greater details in sec. 2, and demonstrate the usability of our approach with real-time 3-D object detection on the HoloLens in sec. 3.

2 Hardware Setup

2.1 System overview

Fig. 1-(a) shows a user wearing our system, which is also illustrated schematically in fig. 2. It is composed of an Intel RealSense

(a) HoloLens 3D (b) RealSense 3D (c) Scene

Figure 3: Comparison of 3D data obtained from the HoloLens (a) and the RealSense (b) for the same scene (c). The 3D HoloLens data is insufficient for many applications such as small scale object detection. For clarity, the background has been cropped out in the 3D data for both (a) and (b). Please see the supplementary video for an animated version of this figure [2].

RGBD camera that is attached to a Microsoft HoloLens headset via a custom-made mount. The RealSense is connected to an Intel Compute Stick PC (specifically, Intel Core M5-6Y57 at 1.1 GHz with 4GB RAM) via USB 3.0. The stick PC is attached to the back of the HoloLens, and the battery pack can typically be carried in the users pockets. Thus, our system does not sacrifice mobility and is still completely tetherless.

Real-time depth data obtained from the RealSense camera can be processed by the PC, which can beam it back to the HoloLens directly via WiFi. To limit bandwidth usage however, it is typically preferable to have the PC implement the particular task at hand and send the results back to the HoloLens, rather than transmitting the raw depth data. The specific task and transmitted data depend upon the application: in sec. 3 we demonstrate the use of our system in the context of small-scale object detection, but others could be used (e.g. depth-based face detection).

2.2 Calibration

This section describes the calibration procedure to transform the depth data in the RealSense reference frame to the HoloLens reference frame so that it can be displayed accurately to the user. Unfortunately, as shown in fig. 3, the depth data provided by the HoloLens is too coarse to be useful for calibration. Therefore, we must rely on the color cameras also present to do so.

Fig. 4 provides an overview of the transformations that must be computed to display the RealSense depth, originally in the "Depth" coordinate system, in the HoloLens virtual camera coordinate system "Virtual". This amounts to computing the transform $\mathbf{T}^{\text{Virtual}}_{\text{Depth}}$ (in our notation, $\mathbf{T}^b_a$ is the transformation that maps a point $\mathbf{p}_a$ in reference frame a to reference frame b, i.e. $\mathbf{p}_b = \mathbf{T}^b_a \mathbf{p}_a$). Following fig. 4 and chaining the transformations, we have:

$$\mathbf{T}^{\text{Virtual}}_{\text{Depth}} = \mathbf{T}^{\text{Virtual}}_{\text{Webcam}} \mathbf{T}^{\text{Webcam}}_{\text{RGB}} \mathbf{T}^{\text{RGB}}_{\text{Depth}} , \quad (1)$$

where "RGB" and "Webcam" are the RealSense color camera and the HoloLens webcam coordinate systems respectively.

We must therefore estimate the three individual transformations in (1). First, we directly employ the transformations $\mathbf{T}^{\text{RGB}}_{\text{Depth}}$ and $\mathbf{T}^{\text{Virtual}}_{\text{Webcam}}$ that are provided by the RealSense and HoloLens APIs respectively. Then, we estimate the remaining $\mathbf{T}^{\text{Webcam}}_{\text{RGB}}$ by placing a planar checkerboard calibration target (which defines the "Calib" reference frame) that is visible by all cameras simultaneously in the scene, and estimating the individual transformations $\mathbf{T}^{\text{Calib}}_{\text{RGB}}$ and $\mathbf{T}^{\text{Calib}}_{\text{Webcam}}$ with standard camera calibration (we use the OpenCV implementation of [10]). Finally, $\mathbf{T}^{\text{Webcam}}_{\text{RGB}} = (\mathbf{T}^{\text{Calib}}_{\text{Webcam}})^{-1} \mathbf{T}^{\text{Calib}}_{\text{RGB}}$. Please refer to fig. 4 for a graphical illustration of this process.

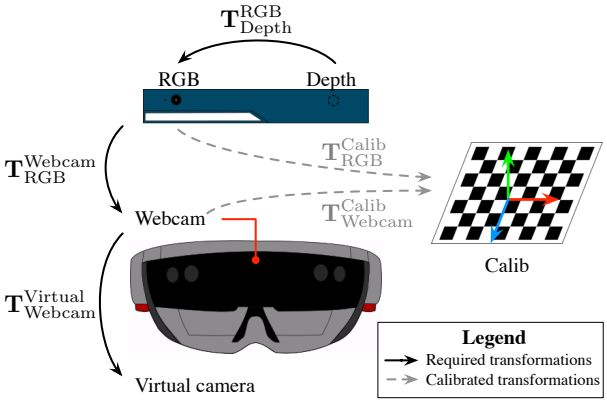

Figure 4: Several rigid transformations must be estimated in order to express the 3D information acquired by the RealSense depth sensor in the HoloLens virtual camera coordinate system. We employ a checkerboard pattern to determine the relationship between the color cameras, since the HoloLens depth information is not reliable enough to obtain accurate calibration results. Bold lines and transformations indicate what is required, and the grayed ones indicate what we explicitly calibrate.

3 Real-time 3D object detection on HoloLens

By comparing the 3D data provided by the HoloLens to the one by the RealSense in fig. 3, it is easy to see that the HoloLens data is insufficient for applications that require high resolution depth data. One such application is object detection and pose estimation, where the task is to accurately locate a known object in the scene.

To demonstrate the usefulness of our system, we thus implemented the multimodal detection system proposed in [4] to detect candidate objects and estimate their corresponding poses. Similarly to [3], we train a template-based detector from a 3D CAD model. The detections are then refined through a photometric verification with the projected 3D model, followed by a fast geometric verification to remove false positives. The most confident object pose are subsequently refined through a dense and more accurate geometric verification step with ICP. This procedure is repeated independently at every frame (although tracking [8] could also be employed to obtain more stable results under object or camera movement).

Fig. 5 shows results obtained on three different, small-scale objects. The object detection and pose estimation algorithms are run on the stick PC, and only the 6-DOF pose information is transmitted back to the HoloLens (see fig. 2). The high resolution mesh of the corresponding object, initially pre-computed and pre-loaded on the HoloLens, is then transformed according to the detected pose

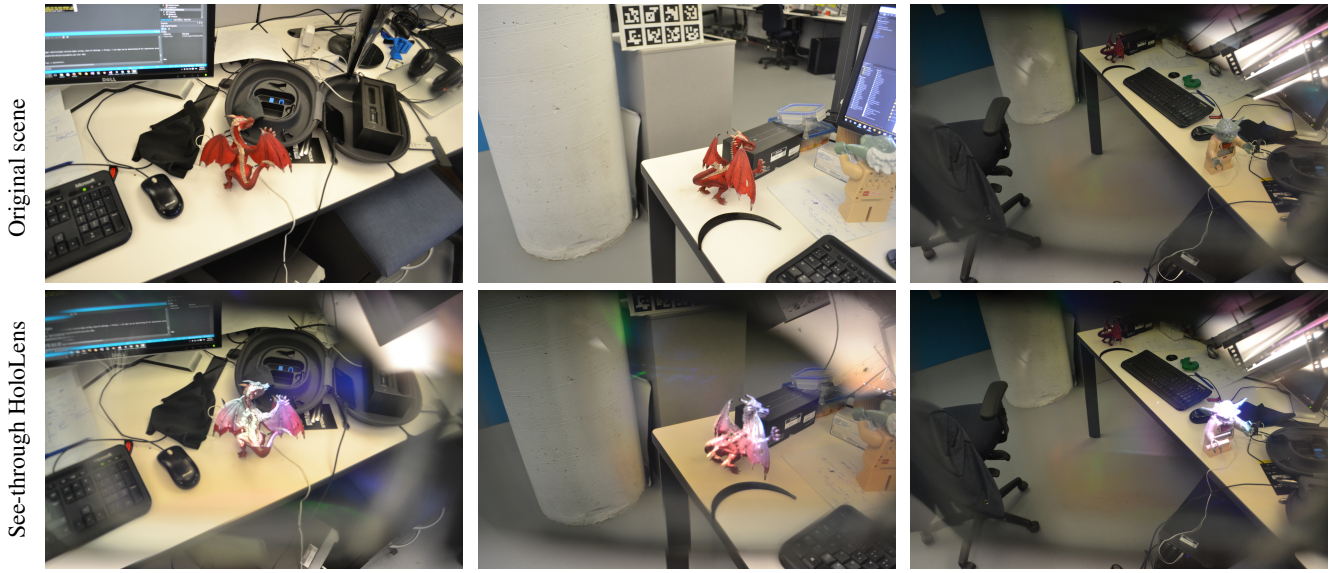

Figure 5: Object detection results. Original scenes are shown on the top row. On the bottom row, the corresponding scene is photographed by placing the HoloLens over the camera to simulate what a viewer would see. A high resolution mesh of objects detected via the high resolution depth stream are realistically overlaid on the scene. Please see the supplementary video for an animated version of this figure [2].

and overlaid at the correct location in the HoloLens display.

4 Discussion

In this poster, we presented a simple system for providing HoloLens applications with real-time and high resolution 3D data. We do so by attaching another RGBD camera—the Intel RealSense in our case but others could be used as well—to the HoloLens, and by streaming the acquired data (or post-processed information) back to the HoloLens via a stick PC. We demonstrate the usefulness of our system by detecting small objects with the high resolution 3D data and overlaying their 3D model in the HoloLens viewpoint, which would be impossible to do from the original 3D data.

In future work, we will improve the precision of the system by explicitly measuring the latency caused by communication and the data acquisition rates of the RealSense. Our current solution works well even without latency compensation, but fast camera or object motions may cause misalignments between the detections and the real objects. We believe that accounting for latency will be an important next step in the development of even more stable systems.

Acknowledgements

This project was supported by Frima Studio and Mitacs through an internship to Mathieu Garon. We also thank Frima Studio for the use of their equipment and facilities.

References

[1] A. Dai, M. Nießner, M. Zollhöfer, S. Izadi, and C. Theobalt. BundleFusion: Real-time globally consistent 3D reconstruction using online surface re-integration. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[2] M. Garon, P.-O. Boulet, J.-P. Doiron, L. Beaulieu, and J.-F. Lalonde. Real-time high resolution 3d data on the hololens: Project webpage. http://vision.gel.ulaval.ca/~jflalonde/projects/hololens3d/, July 2016.

[3] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, K. Konolige, and N. Navab.

[4] S. Hinterstoisser, S. Holzer, C. Cagniart, S. Ilic, K. Konolige, N. Navab, and V. Lepetit. Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In *IEEE International Conference on Computer Vision*, 2011.

[5] R. Newcombe, D. Fox, and S. M. Seitz. DynamicFusion: Reconstruction and tracking of non-rigid scenes in real-time. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[6] C. Y. Ren, V. Prisacariu, O. Kaehler, I. Reid, and D. Murray. 3D tracking of multiple objects with identical appearance using RGB-D input. In *International Conference on 3D Vision*, 2015.

[7] R. Rios-Cabrera and T. Tuytelaars. Discriminatively trained templates for 3D object detection: A real time scalable approach. In *IEEE International Conference on Computer Vision*, 2013.

[8] D. J. Tan, F. Tombari, S. Ilic, and N. Navab. A versatile learning-based 3D temporal tracker: Scalable, robust, online. In *IEEE International Conference on Computer Vision*, 2015.

[9] P. Wohlhart and V. Lepetit. Learning descriptors for object recognition and 3D pose estimation. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[10] Z. Zhang. A flexible new technique for camera calibration. *IEEE Transactions on pattern analysis and machine intelligence*, 22(11):1330–1334, 2000.