Real-Time Depth Sensing by Combining Computer Vision and LIDAR Point-Cloud Data

Kin-Ho Lam
College of Engineering
Oregon State University
kinholam5@gmail.com

May 22, 2018

Abstract

Modern self-driving car systems face a massive challenge: creating a holistic system capable of accurately identifying obstacles and calculating their distances in real time. Such challenges are nontrivial because one must consider multiple conflicting sensory interferences. Ambient light from the Sun, reflective surfaces, darkness, fog, and other sensory obsurements pose significant challenges to guaranteeing an autonomous driving system's safety. Computer vision machine learning models are capable of object recognition and depth perception, but only to a degree of certainty. LIDAR (Light Imaging Detection and Ranging) sensors are capable of high-accuracy depth sensing, but are affected by fog and are only able to collect point-cloud data in a single plane. This project demonstrates real-time object detection and depth sensing is feasible by combining a computer vision model and live LIDAR point-cloud data through using simple geometry.

Video Demo: https://youtu.be/6v8h8LPFRls
Project Source: https://github.com/DSCVL

***K*eywords** Depth sensor · LIDAR · Computer Vision

1 Introduction

1.1 Existing Depth Sensing Technologies

Current depth sensing technologies cannot singularly serve as the only sensory input of an autonomous self-driving system capable of accurately identifying obstacles and quickly calculating their distances in real time.

1.1.1 Radar

Radar is an object-detection system that uses radio waves to determine the distance, angle, and velocity of objects relative to the sensor. Radar works by sending out electromagnetic waves, then measuring the intensity of the reflection. A radar system consists of a transmitter producing electromagnetic waves, a transmitting antenna, a receiving antenna, and a processing system to interpret the data and determine the locations of objects. Such systems are widely used for military applications, airplanes, and weather forecasting. While radar systems have been tested to work reliably in extreme weather conditions, unlike LIDAR, radar cannot easily detect small objects. This is problematic when applying a radar to an autonomous system; a radar system may not detect a narrow pole or small object in front of it.

1.1.2 Infrared Sensors

IR, or infrared depth sensors, strobe an infrared pattern on objects infront of it. **??** This infrared pattern is picked up by a receiving camera, which uses the pattern and some geometric algorithms to calculate distance. While accurate and cheap, IR depth sensors are not suited for autonomous systems because they are sensitive to variable light conditions.

Additionally, two IR depth sensors pointing at the same subject will overlap and confuse each other's sensors. Natural sunlight will also wash out the IR pattern and blind the sensor.

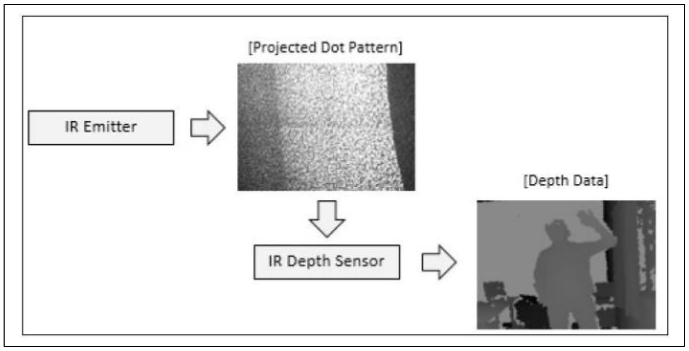

Figure 1: Kinect depth sensor translating the IR-dot pattern into a 3D representation.

1.1.3 LIDAR

LIDAR (light detection and ranging) sensors uses a laser and its returning time and angle of incidence to determine distance objects relative to the sensor. Most commercial LIDAR devices use a low powered 600-1000 nm laser which is mounted on a motor to pan the laser capturing a single plane. A receiver is used to measure the time it takes for the laser to bounce back to the sensor upon encountering an object. Their ability to work in varying lighting conditions and ability to detect small objects makes them well suited for use in self-driving cars or autonomous robots. However their limitation to a single plane necessitates other complimentary sensors.

2 Design

The Logitech Brio webcam provides a high-resolution, two-dimensional image but lacks depth perception. The LIDAR provides accurate depth measurement but can only capture point cloud data in a single plane. This project proposes bridging the utility of both devices by securing them in stationary positions, then using software to combine their measurements. This involves using the M16 LIDAR to get depth sensing information and using computer vision to recognize objects. The result is a scalable and reliable depth sensor that will not be affected by natural light, and can be further improved by training a better computer vision model or adding more sensors. This project hopes to achieve a proof of concept design to be showcased in a live demo at Oregon State University's 2018 Undergraduate engineering expo. This live demo shall consist of the full system pointed at the project booth's audience.

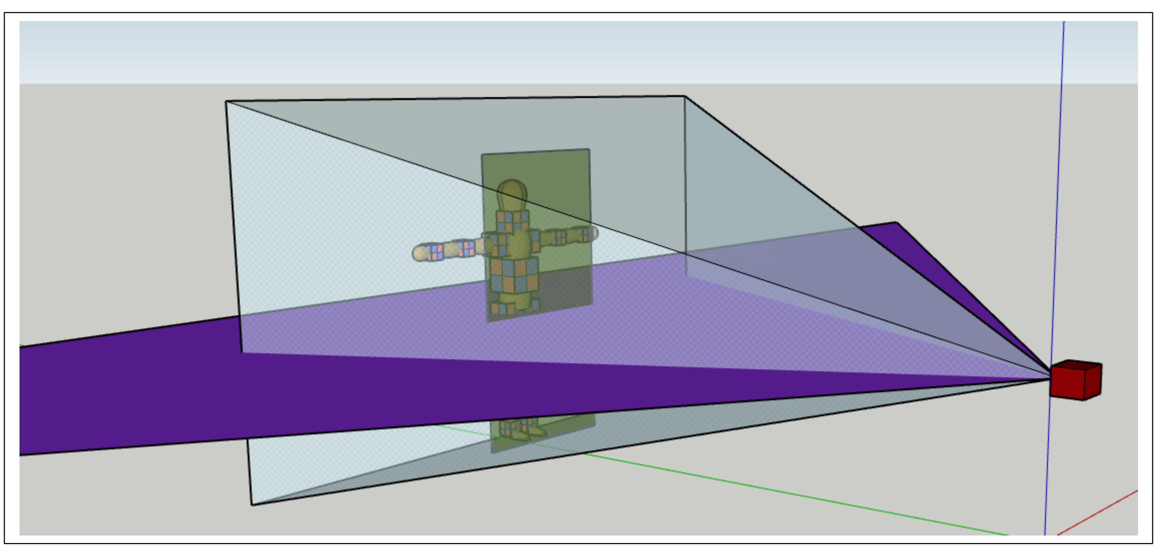

Figure 2: Visualizing different dimensions measured by the LIDAR and Brio Webcam.

Figure **??** illustrates different dimensions measured by the M16 LIDAR and Brio Webcam. The red cube represents the Logitech Brio webcam and M16 LIDAR secured in stationary positions. The flat purple triangle represents the M16 LIDAR's horizontal range detection. The transparent green rectangle in front of the person represents the computer vision model recognizing that there is a person in-front of the sensor. The transparent teal pyramid represents the Brio webcam's field-of-view.

2.1 Matching Point Cloud Data to Computer Vision Output

I developed a work-around to compensate for the slow LIDAR device by splitting the RPLIDAR A1 readings into a seperate thread that pushes data into a shared buffer. This allows the Tensorflow model to poll the buffer for new distance data as it needs it. However, this work around doesn't completely solve the rotational bottleneck issue. I observed the system's video output dropping to about 25 frames per second.

Aligning the RPLIDAR A1's laser sweep with the webcam's field of view was a challanging task. Upon detecting an object, the RPLIDAR A1 returns several points of incidence consisting of distance and the object's angle with respect to the tip of the device. Figure **??** illustrates the RPLIDAR A1's measurements as a polar plot. The tip of the device is considered 0^o, this is an important reference because it splits the usable areas into two hemispheres. Through trial and error, I discovered that while in the mount, the RPLIDAR A1's usable area ranged from $0^o - 55^o$ and $0^o - 305^o$, illustrated as a green triangle in figure **??**.

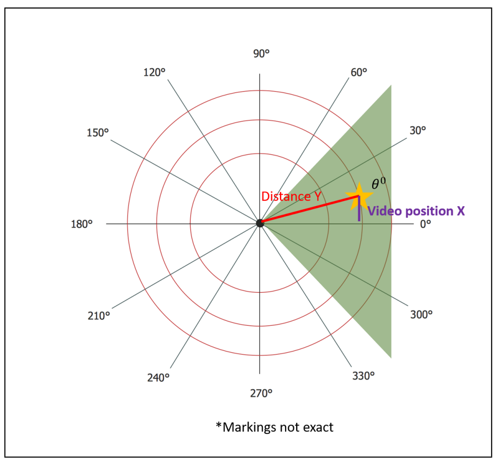

Figure 3: Polar illustration of the RPLIDAR A1's detection.

One can now use this angle to transpose a point in X-Y dimensions into X-Z video dimensions via similar triangles geometry. Let the yellow star in figure **??** represents a point of incidence that the RPLIDAR A1 detects is at angle θ. One can now use the following equations to translate this point onto the video.

$$\theta \leq 55 : X = \frac{videowidth}{2} * sin(\theta) + \frac{videowidth}{2} \tag{1}$$

$$\theta \geq 305 : X = \frac{videowidth}{2} * sin(360 - \theta) \tag{2}$$

Adding an offset $\frac{videowidth}{2}$ is necessary because the video output is mirrored as shown in figure **??**.

2.1.1 Headings: third level

Morbi pharetra aliquam diam vel tincidunt. Nunc eget dui dictum, scelerisque dolor suscipit, finibus orci. In hac habitasse platea dictumst. Sed consequat facilisis nibh, quis porta neque iaculis a. Nullam libero ipsum, facilisis at arcu vitae, accumsan placerat dolor. Duis non viverra urna, in consequat nisl. Aenean ut purus quis nisl efficitur mattis. Nulla facilisi. Etiam nec gravida erat. Sed tempus, felis in tincidunt eleifend, massa leo facilisis odio, nec ornare sapien diam vitae massa. Maecenas porttitor auctor felis, ut venenatis dui venenatis et. Mauris tincidunt, erat quis euismod tempor, velit arcu pharetra orci, eu eleifend nulla risus ac neque. Cras dapibus libero volutpat diam sollicitudin eleifend.

Paragraph Sed faucibus diam eget augue consectetur, fermentum tempor urna efficitur. Donec vitae tempus nulla, ut rhoncus purus. Maecenas fermentum vehicula turpis ac luctus. Suspendisse potenti. Fusce est sapien, rhoncus et fermentum vitae, pulvinar sit amet metus. Proin sollicitudin venenatis ante, vitae consectetur purus feugiat eget. Mauris efficitur, sapien nec ullamcorper tempus, erat tellus ultrices felis, a auctor nisi ipsum eget lorem. Cras tristique quis metus et mollis. Cras interdum neque id diam eleifend tristique. Ut turpis leo, molestie a ex et, semper semper lorem. Integer cursus enim velit, in vehicula nibh condimentum non. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Etiam commodo finibus neque, a consectetur tellus facilisis vitae. Praesent lacinia auctor ligula ut scelerisque. Nulla facilisi. Duis placerat lorem odio, sed volutpat metus pellentesque egestas.

3 Examples of citations, figures, tables, references

Quisque elit eros, fringilla efficitur sapien in, scelerisque egestas augue. Suspendisse commodo ullamcorper mi, a tempus mi blandit vel. Suspendisse eleifend dolor laoreet dolor efficitur, et interdum arcu interdum. Phasellus nisl metus, malesuada in dapibus a, tincidunt sed lacus. Suspendisse feugiat semper venenatis. Donec nec augue facilisis, porttitor erat id, facilisis odio. Duis rutrum pretium maximus. Vivamus quis justo et mi ullamcorper cursus. Aenean sed quam laoreet, semper arcu eget, blandit lacus. Aenean non elementum quam. Proin eu sodales sapien. Aliquam sodales lectus quis placerat sodales. [**?**, **?**] and see [**?**].

The documentation for `natbib` may be found at

`http://mirrors.ctan.org/macros/latex/contrib/natbib/natnotes.pdf`

Of note is the command `\citet`, which produces citations appropriate for use in inline text. For example,

```
\citet{hasselmo} investigated\dots
```

produces

> Hasselmo, et al. (1995) investigated...

`https://www.ctan.org/pkg/booktabs`

3.1 Figures

Curabitur imperdiet urna finibus diam ultricies mollis. Mauris at felis in nisi imperdiet convallis. Fusce viverra aliquam cursus. Nullam eget cursus tellus, at tincidunt arcu. Praesent felis urna, fermentum ac turpis sed, tempus gravida ex. Mauris pretium ante maximus augue ullamcorper, iaculis vestibulum mi scelerisque. Nulla nec ornare sem. Aliquam eu ante eu arcu volutpat hendrerit. Donec a neque molestie, iaculis turpis sed, semper sem. Donec non luctus urna, vitae ultrices nisi. Aenean iaculis interdum sem non ultricies. Vestibulum eleifend turpis eu quam tristique, sed eleifend urna aliquet. See Figure **??**. Here is how you add footnotes. [1]

Aenean ut leo eget urna cursus scelerisque. Phasellus vitae scelerisque mi. Nam sed eros scelerisque, tempus ligula vitae, auctor nisi. In et augue imperdiet, placerat neque eu, pretium est. Suspendisse malesuada nec arcu vitae venenatis. Mauris maximus iaculis finibus. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. In mattis molestie magna, ac mattis purus volutpat ac. Integer non lacus vel elit ullamcorper laoreet. Pellentesque a hendrerit elit.

3.2 Tables

Donec dignissim nibh lacinia dolor dignissim luctus. Pellentesque pellentesque ultricies ipsum, sit amet varius lectus feugiat ut. Suspendisse potenti. Nullam vel sodales diam. Nam sollicitudin gravida elit, non scelerisque nunc blandit vel. Proin arcu est, consequat eu ex viverra, luctus rutrum nibh. In mollis leo et eros sollicitudin, et imperdiet dolor gravida. Ut sem velit, viverra vitae lacus in, condimentum viverra orci. Duis ultrices volutpat malesuada.

See awesome Table **??**.

3.3 Lists

- Lorem ipsum dolor sit amet

[1] Sample of the first footnote.

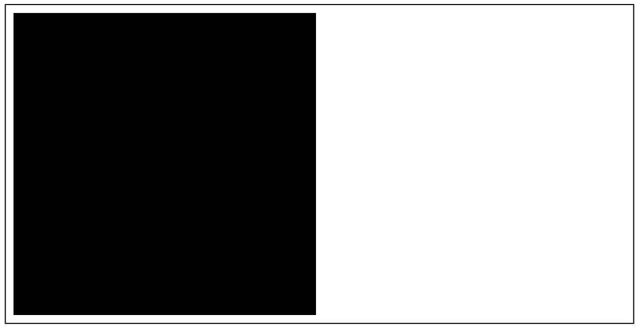

Figure 4: Sample figure caption.

Table 1: Sample table title

Part		
Name	Description	Size (μm)
Dendrite	Input terminal	$\sim$100
Axon	Output terminal	$\sim$10
Soma	Cell body	up to 10^6

- consectetur adipiscing elit.
- Aliquam dignissim blandit est, in dictum tortor gravida eget. In ac rutrum magna.

References

[1] George Kour and Raid Saabne. Real-time segmentation of on-line handwritten arabic script. In *Frontiers in Handwriting Recognition (ICFHR), 2014 14th International Conference on*, pages 417–422. IEEE, 2014.

[2] George Kour and Raid Saabne. Fast classification of handwritten on-line arabic characters. In *Soft Computing and Pattern Recognition (SoCPaR), 2014 6th International Conference of*, pages 312–318. IEEE, 2014.

[3] Guy Hadash, Einat Kermany, Boaz Carmeli, Ofer Lavi, George Kour, and Alon Jacovi. Estimate and replace: A novel approach to integrating deep neural networks with existing applications. *arXiv preprint arXiv:1804.09028*, 2018.