CS766 Project Midterm Report

Dongning Wang
Mar 29, 2016

Background

Assume in 2D world we are given a camera that can shoot and receive light at sufficiently high speed. The range of view of the camera is known. There are several walls around the camera. Light will be scattered to all directions with uniform probability once reach a wall surface (i.e Lambertian). Time-of-flight information will be recorded if light is bounced back into the camera. As shown in Fig1 below, A, B is within the range of view of the camera, so we can measure the distance from A or B to the camera C by sending light signal to A or B respectively and measure the flight time of the first bouncing back signal. Now assume that we have some way of knowing how many times the light has bounced before captured by the camera. Let's focus on three bounce path first. The path C-A-D-B is a three-bounce path. The middle bounce D lands on a wall which is previously out of the range of view of the camera. By the Time-of-flight information we know the total length of the light path C-A-D-B. Subtract it by AC and BC, we have the length of AD+BD. Note that we still don't know the exact location of D. But it should lie on the ellipse with foci A and B, and the length of long axis 2a = AD+BD. If we repeat the process for another three -bounce path, we will have another ellipse. If we repeat the process for many three-bounce paths, and let the ellipses vote, then the hot spots will recover Wall 2 and Wall 3. This method is called back propagation and is the idea behind [1]. The paper actually applied the method to 3D world and it's currently the state -of-the-art method.

Problem

The purpose in this project is to explore the method, understand its limitation and see wether we can improve its accuracy or extend it to detect walls which can only be reached by higher order bounces, like Wall 4 in Fig.1. If it doesn't work, why? Is it possible to recover Wall 4 with this set up but other methods?

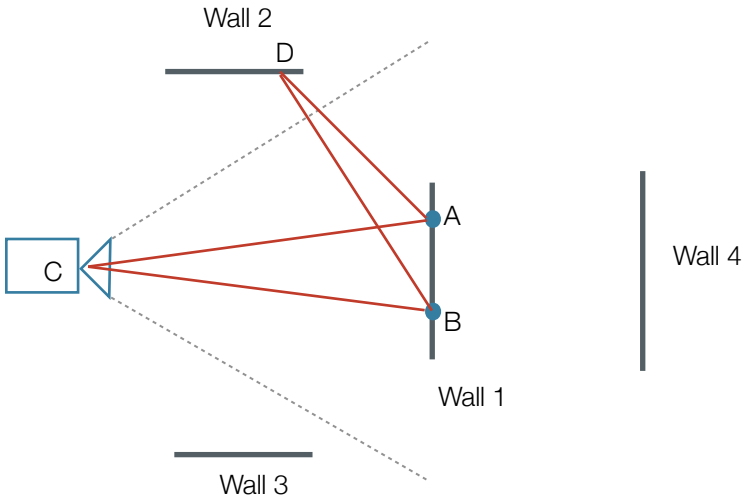

Fig. 1

Current Progress:

We consider the followings as input:

- walls: line segment with start and end points;
- camera location: one point (x,y);
- camera view angles.

A simulator has been built which takes a JSON file containing the above information. It simulates the light paths, and computes the bounces. With the simulator, we have recovered the result in [1]. Fig. 2 is a variation of the configuration in Fig. 1.

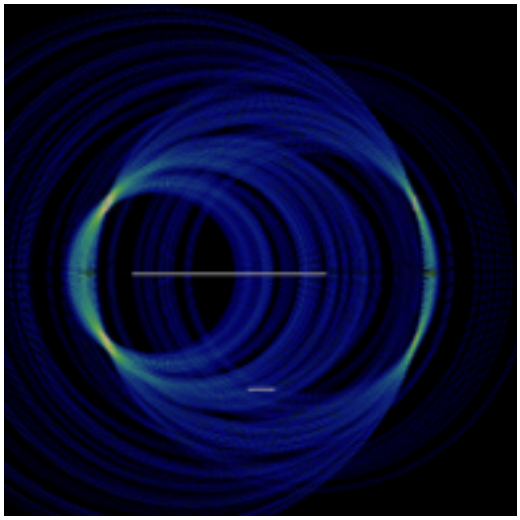

Fig. 2

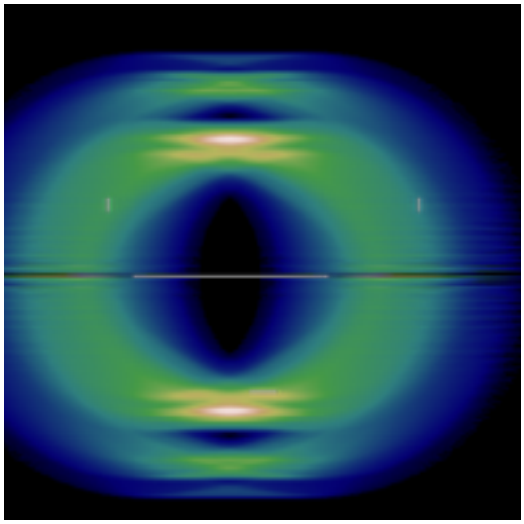

Fig. 3

The difficulty to extend the method to higher order bounce is that we don't know the bouncing location on the walls that are out of the sight of the camera. As a result, when we want to repeat the process to detect Wall 4 after we know Wall 2 and 3 with 5-bounce paths, we don't know which points are the foci of the ellipse. This is similar to the 3-bounce case if we only have the time-of-flight, but don't know where A and B are. In that case a naive method is that for each path, draw ellipses with all possible pairs of points on Wall 1 as foci, and look at the hot spot. The result is shown in Fig. 3. As one can see the information of the two vertical walls are entirely buried by the ambiguity created by Wall 1 (the horizontal one). For the same reason, it will not work for higher order bounce.

Another finding about the method is false detection. The false detection due to the symmetry of ellipses is easy to remove. This will be done by considering the direction of incoming light ray. But the false detection along the line of Wall 1 has a different natural. This happens when Wall 1 is short and therefore the ellipses are too close to each other. This can probably solved by increasing resolution.

We also observed that when the three-bounce detected wall lies transversally to most of the ellipses then the hot spot we got is larger. This motivate us to find a better way to locate the wall rather than using a threshold.

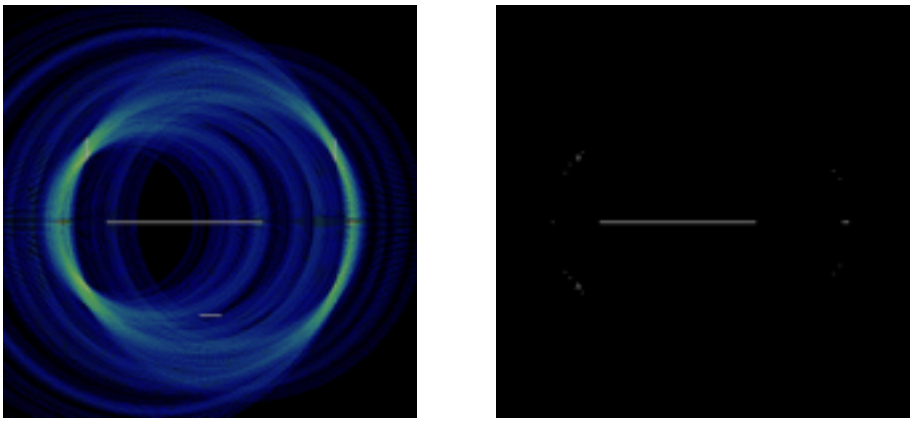

Fig. 4

Next Steps:

To improve the result for three-bounce detection, we can consider the incoming direction of light ray to eliminate some false detection. We will also refactorize the weight of each ellipse in the weight to narrow the hot spot area.

One possible fix of the failure of five-bounce detection is to eliminate the ambient noise imposed by the geometry of Wall 1. In particular, we can subtract the image in Fig. 3 by the image of all possible ellipses with all possible pairs of points on Wall 1 as foci and all possible long axis length.

Another direction is to prove the existence of a solution. Namely the configuration space maps injectively into the signal space. In particular we will start with the configure in Fig.1 but replace Wall 4 with a point. We expect any small perturbation of the point will result in change of the signal function. This will also shed some light to the construction of such solution.

Future Milestones:

April 14th, complete as much items listed above as possible.

April 24th, write report.

Reference:

[1] Andreas Velten, Thomas Willwacher, Otkrist Gupta, Ashok Veeraraghavan, Moungi G. Bawendi & Ramesh Raskar, Recovering three-dimensional shape around a corner using ultrafast time-of-flight imaging. Nature Communications 3, Article number: 745 doi:10.1038/ncomms1747