

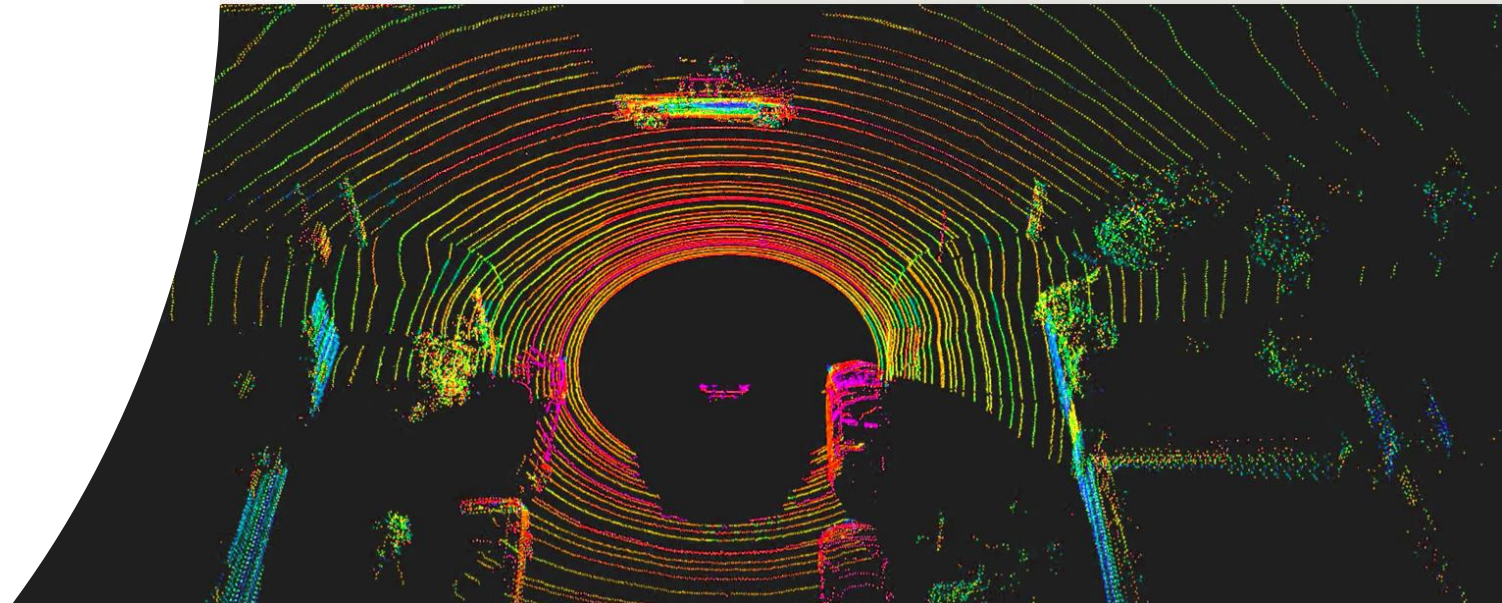
# Light Detection and Ranging (LiDAR)

CS 3630



# LIDAR

- Superpowers:
  - 360 Visibility
  - Accurate depth!
- Almost all AV prototypes have them



[Images and exposition take from Voyage Blog post](#)

<https://news.voyage.auto/an-introduction-to-lidar-the-key-self-driving-car-sensor-a7e405590cff>



# Some Robots that Rely on LIDAR



Digit from Agility Robotics



Quadruped at GT (Ye Zhao's lab)



Jackal from Clearpath Robotics



Spot from Boston Dynamics

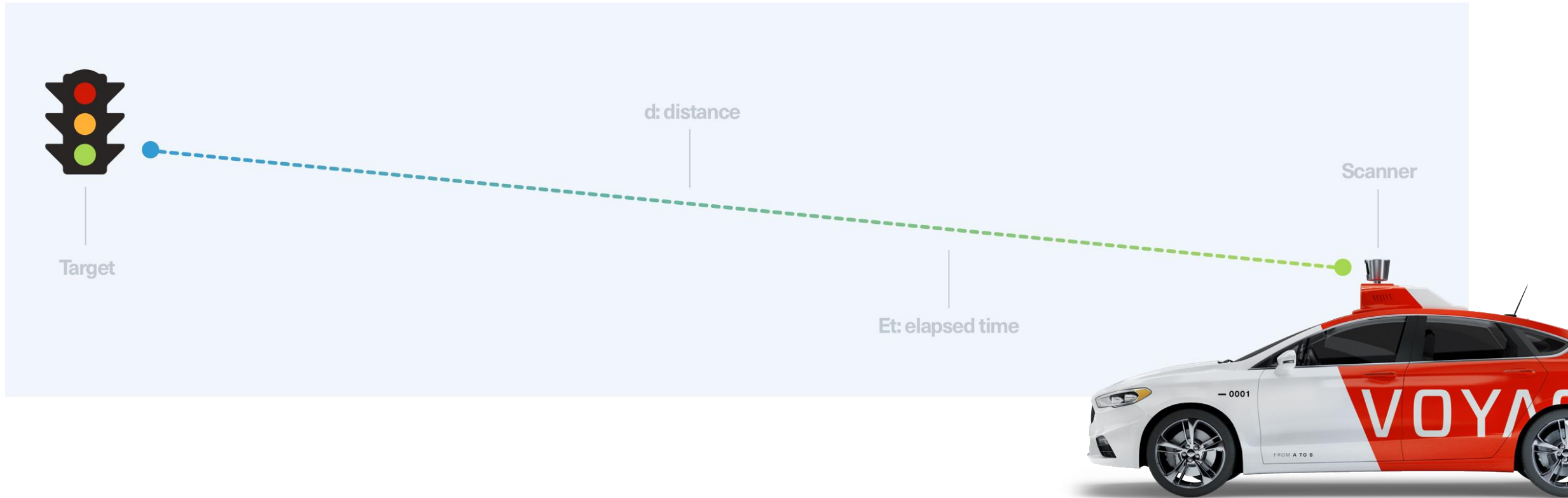


Yellowscan sensor



SIC sensor

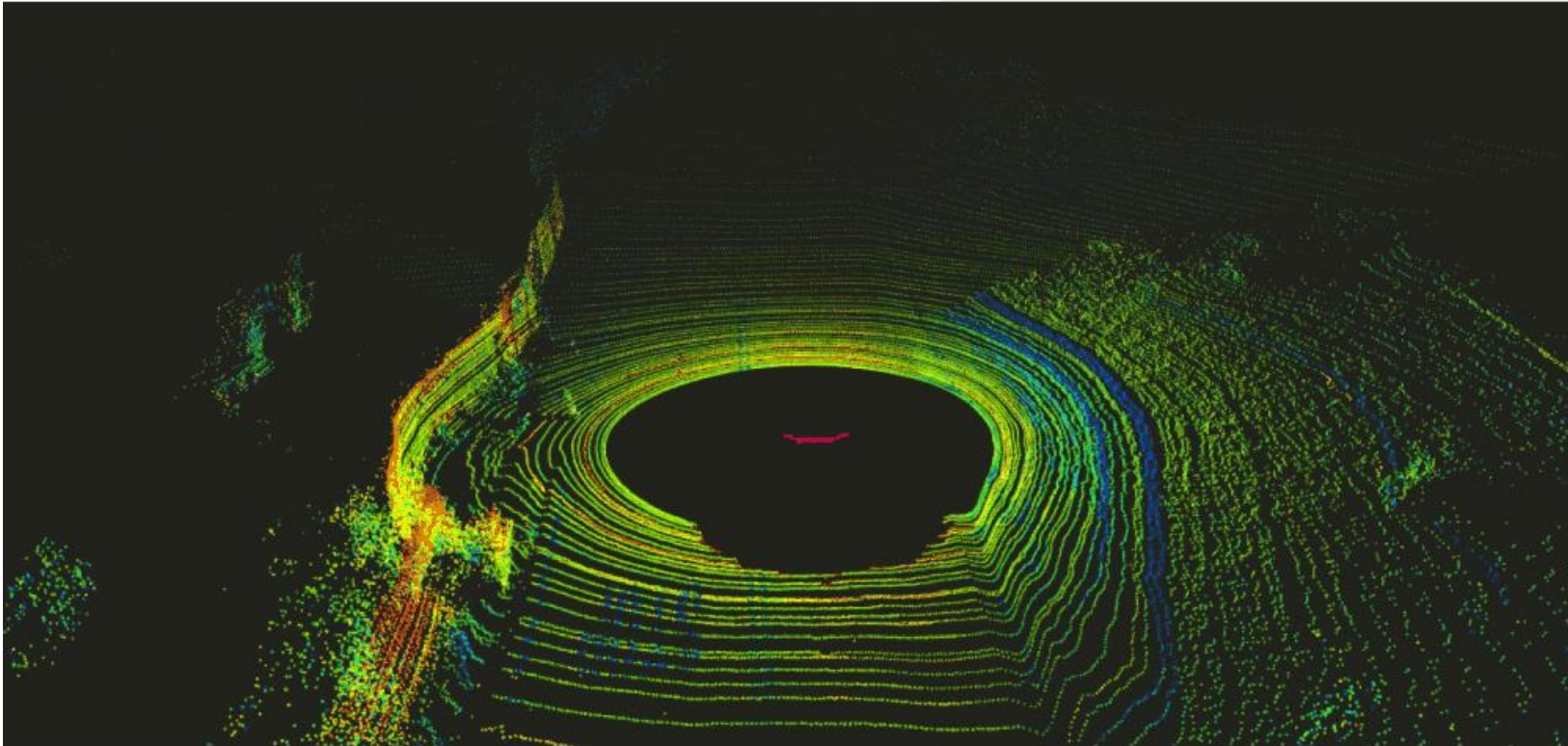
# LIDAR Basic Principle



- Send a light pulse
- Measure elapsed time
- Infer distance

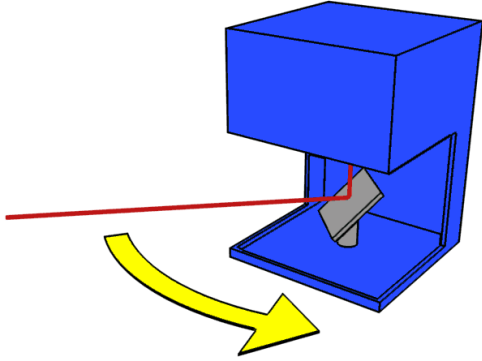
[Images and exposition take from excellent Voyage Blog post](#)

# Example



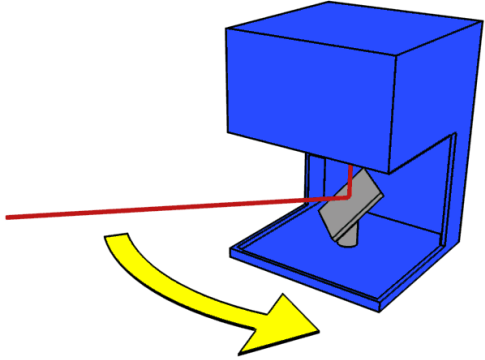
[Images and exposition take from excellent Voyage Blog post](#)

# Basic Idea

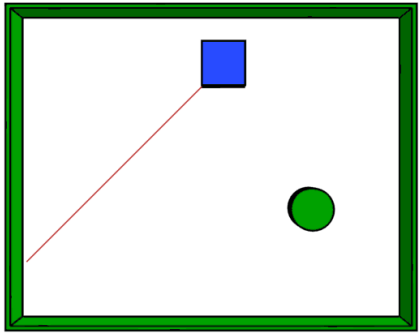


Sweep a laser beam in a circle.  
The beam always lies in a single plane

# Basic Idea

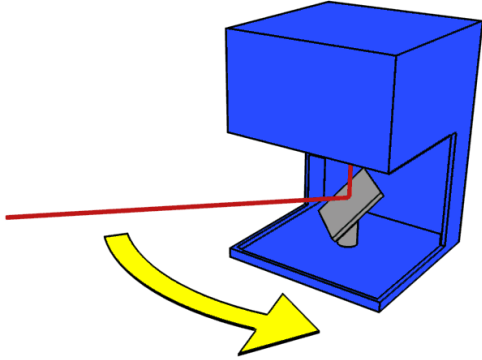


Sweep a laser beam in a circle.  
The beam always lies in a single plane

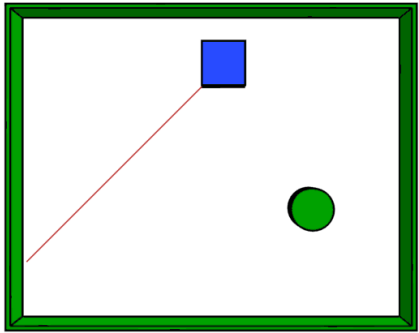


Time of flight of the beam can be measured.  
Time of flight is proportional to distance.

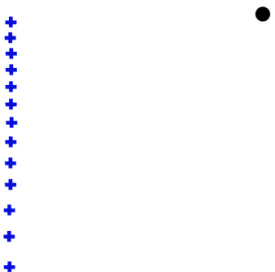
# Basic Idea



Sweep a laser beam in a circle.  
The beam always lies in a single plane



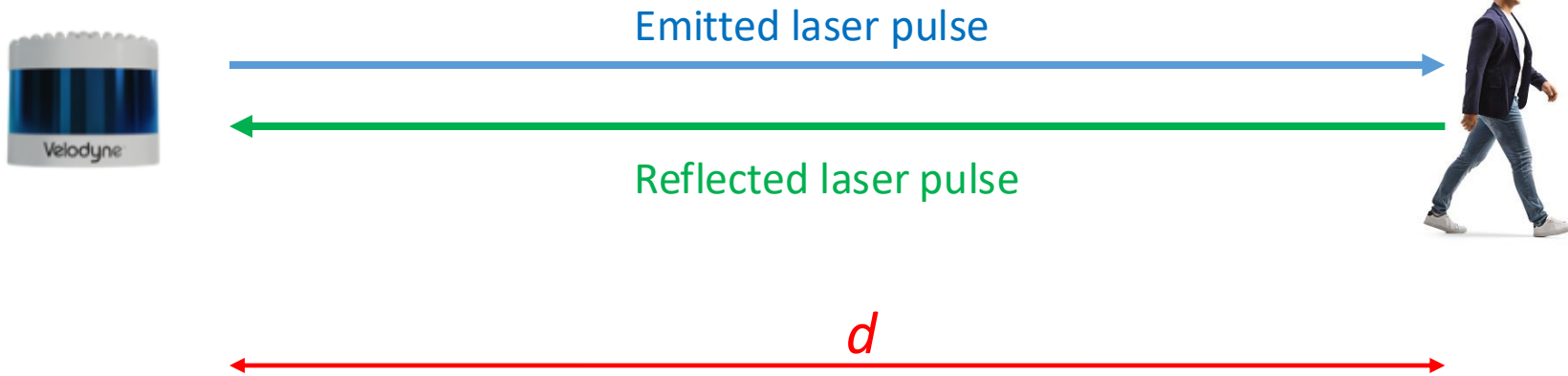
Time of flight of the beam can be measured.  
Time of flight is proportional to distance.



Compute distance at a discrete set of beam angles.  
Once the beam “hits” an object, anything behind the object is occluded.



# Time of Flight



*This assumes that the motion of the pedestrian is slow relative to the speed of light...*

Standard equation: speed  $\times$  time = distance

$$ct = 2d \rightarrow \mathbf{d = 0.5ct}$$

In our case:

- $c$  is the speed of light.
- $t$  is the time from emission to measured return.
- Distance is  $2d$ , since the light travels to the target, and then returns.

## **Example:**

For an object 15 meters from the detector:

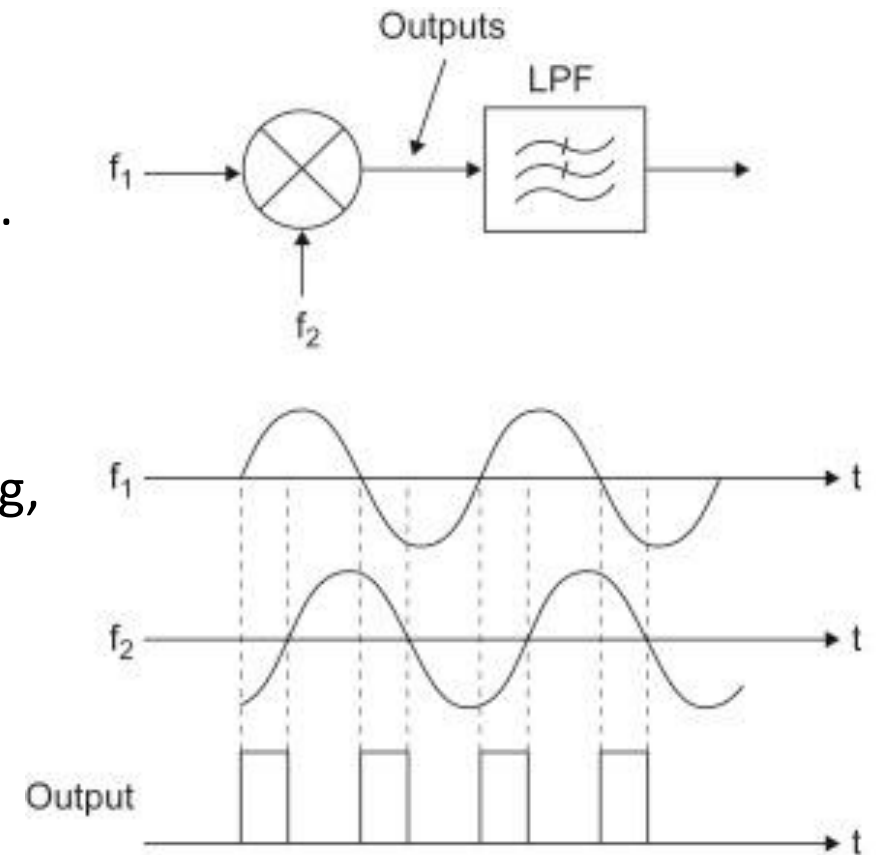
$$0.015 = 300000 \times \frac{t}{2}$$
$$t = \frac{0.030}{300000} = 1 \times 10^{-7} = 0.1 \mu s$$

➤ *We typically don't apply this method literally, because measuring elapsed time between pulses is difficult.*

# Coherent Detection

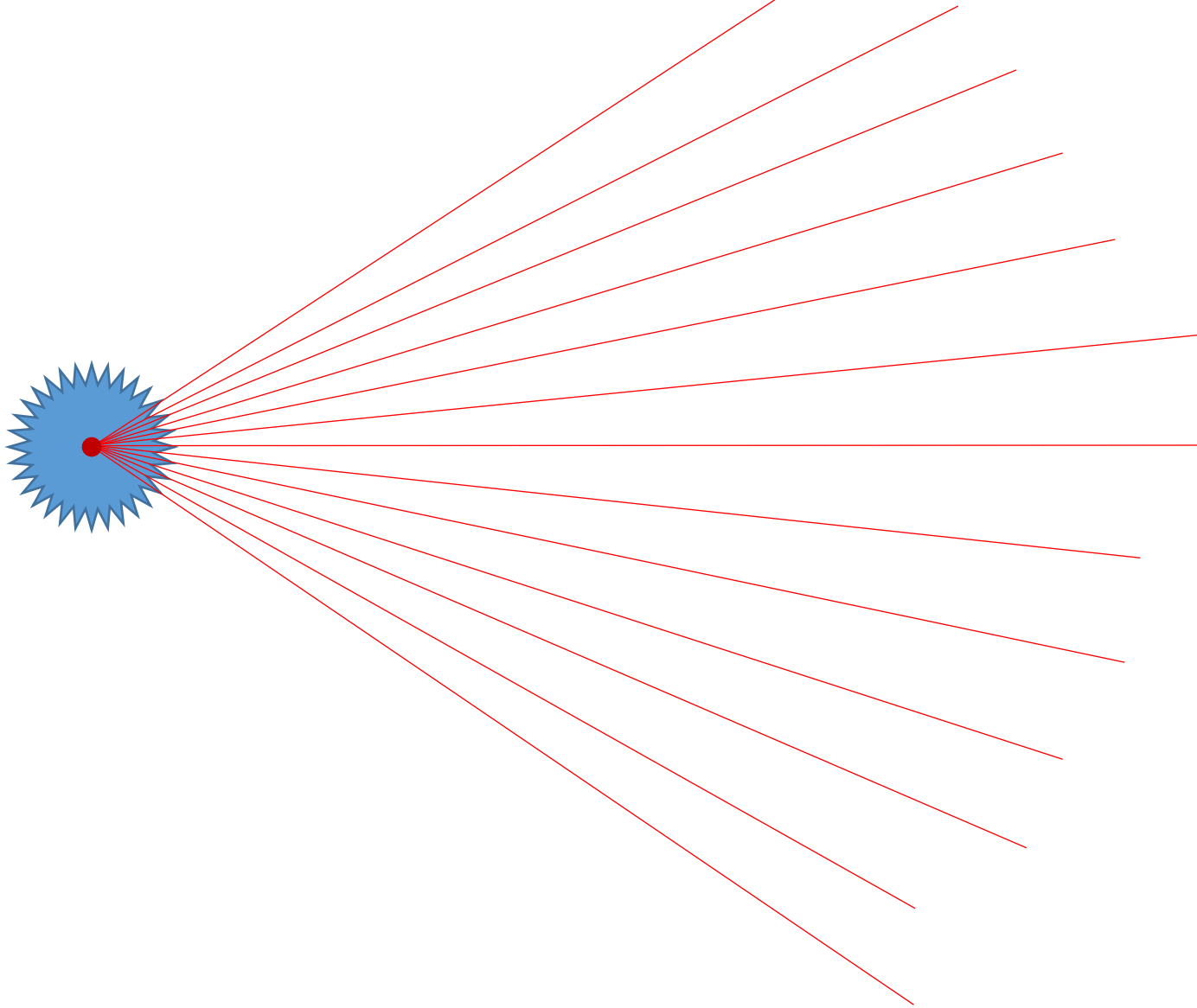
- Instead of measuring time of flight directly for a single pulse...
  - *Emit a periodic signal (e.g., a sine wave), and measure the phase shift between the emitted and received signals.*
- There are plenty of clever circuits that can do this sort of thing, e.g., using a phase locked loop to “lock onto” the received signal.

Here,  $f_1$  is the source signal,  $f_2$  is the received signal, and the width of the pulse is the phase shift, which is equal to the elapsed time of flight.

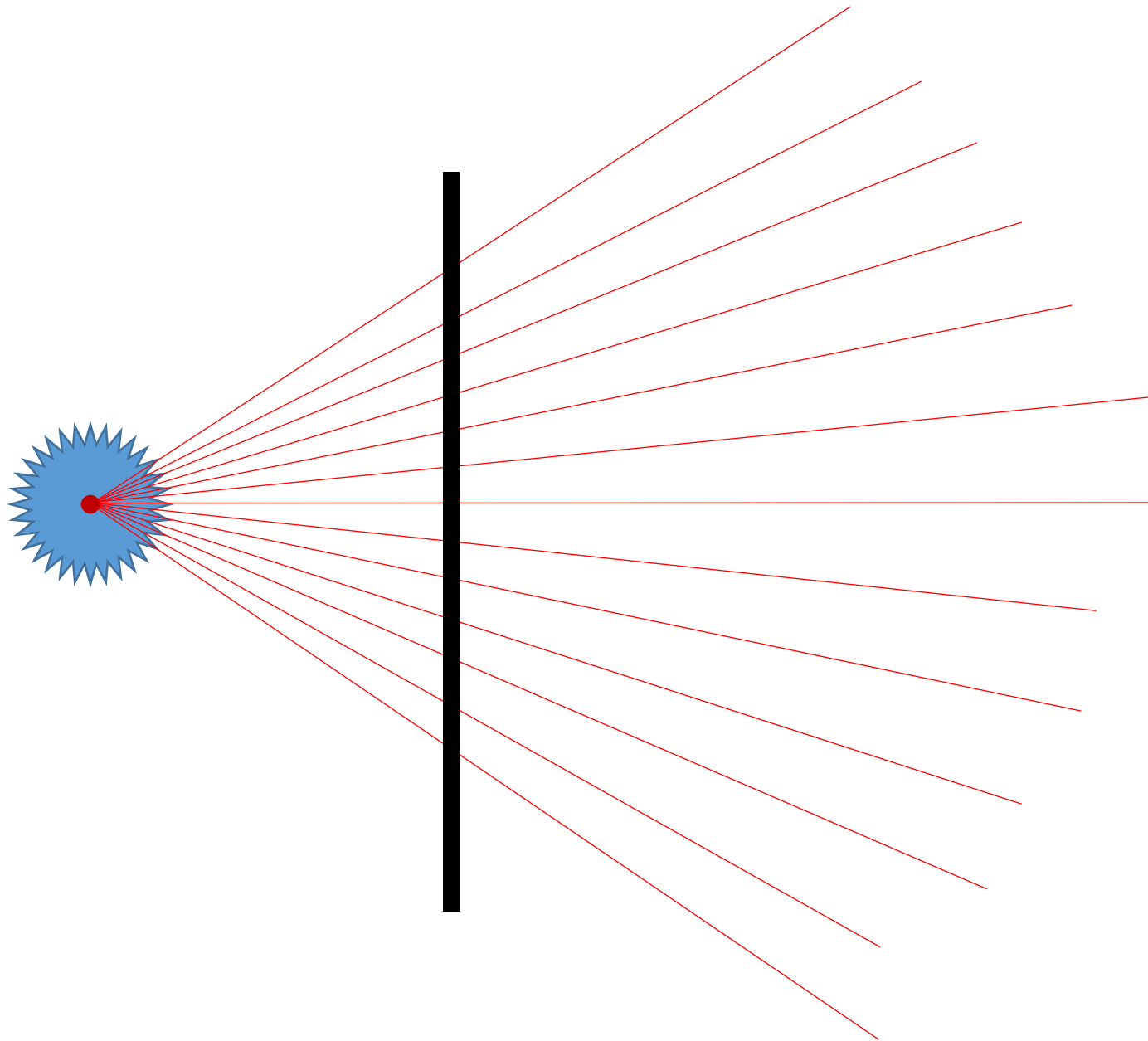


***Zero crossings trigger the output.***

# Equal angular displacements

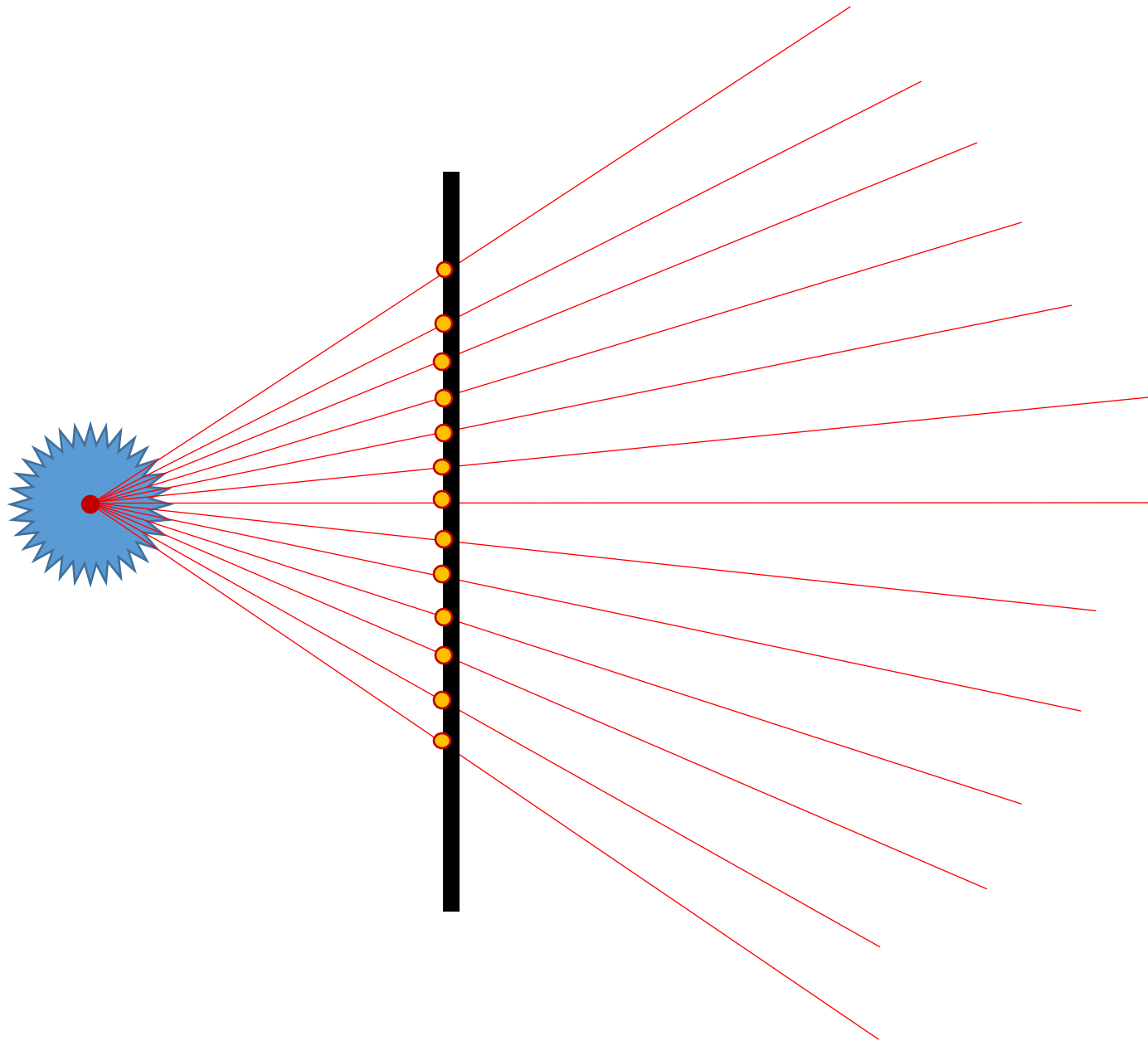


The fact that the measurements are taken at equally spaced orientations does not imply that the measured data (i.e., the point cloud) will be uniformly distributed in space.

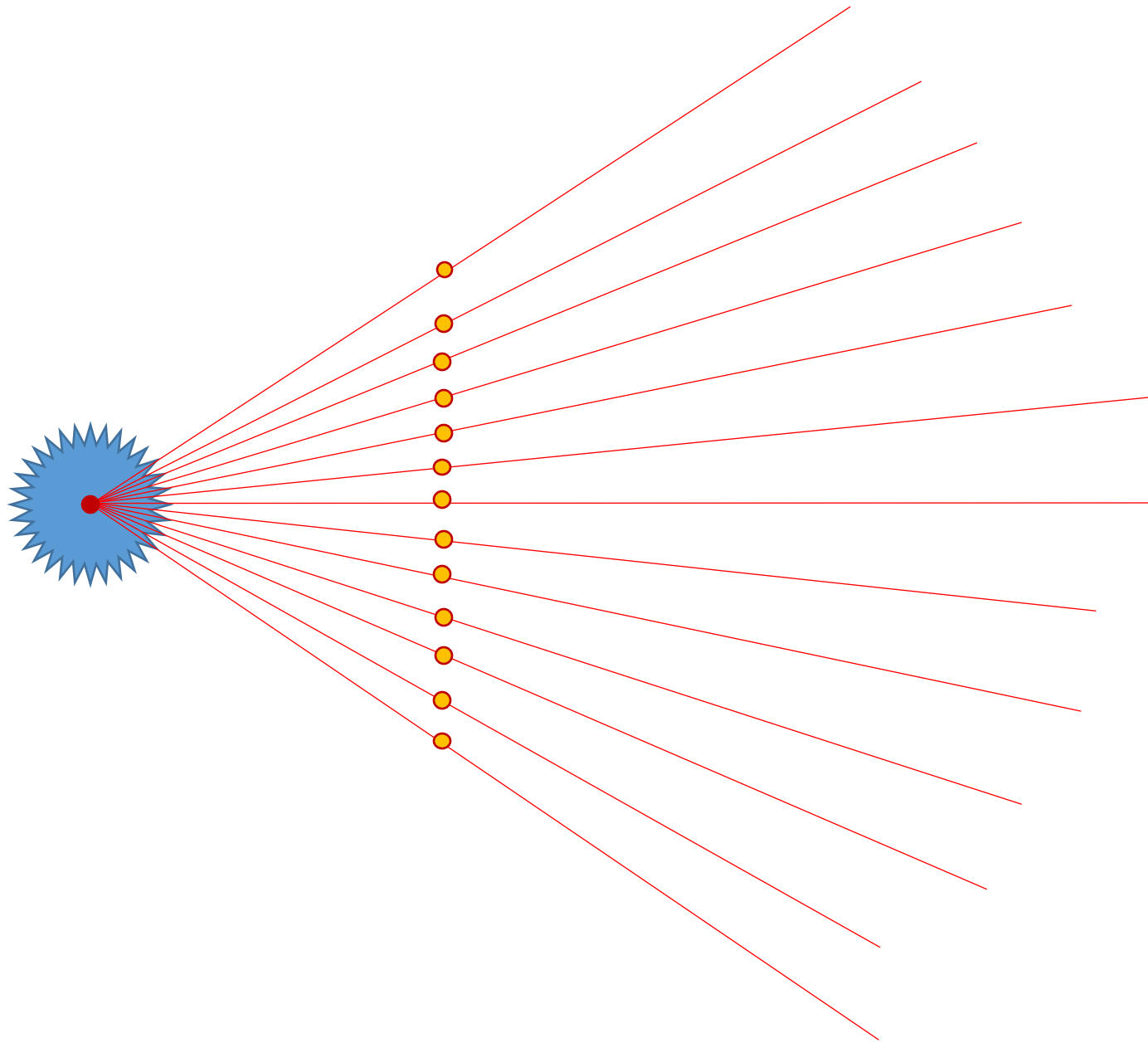


Consider a single long wall.  
The orientation of the wall  
relative to the sensor  
determines the distribution of  
data in the point cloud.



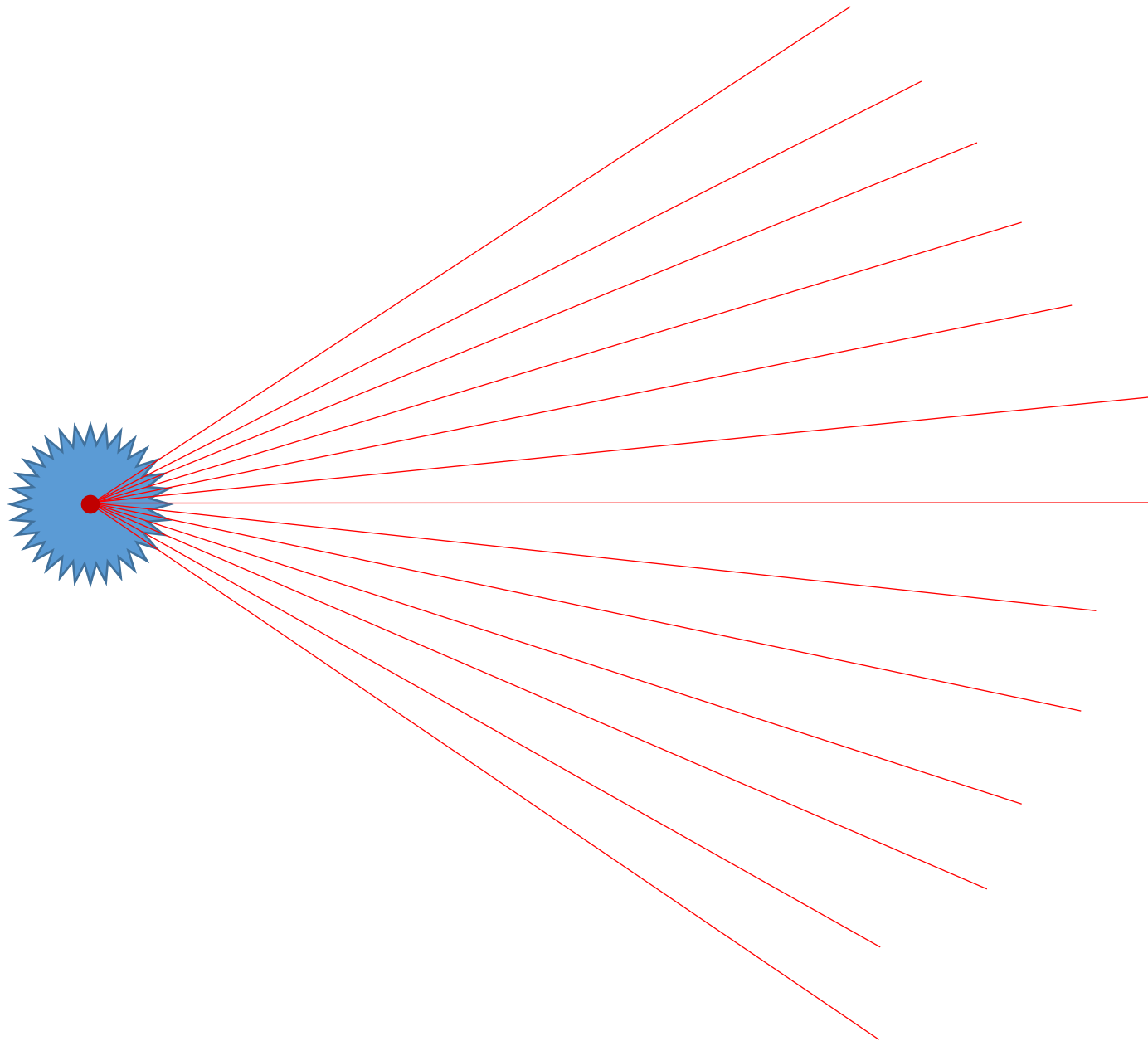


Consider a single long wall.  
The orientation of the wall  
relative to the sensor  
determines the distribution of  
data in the point cloud.

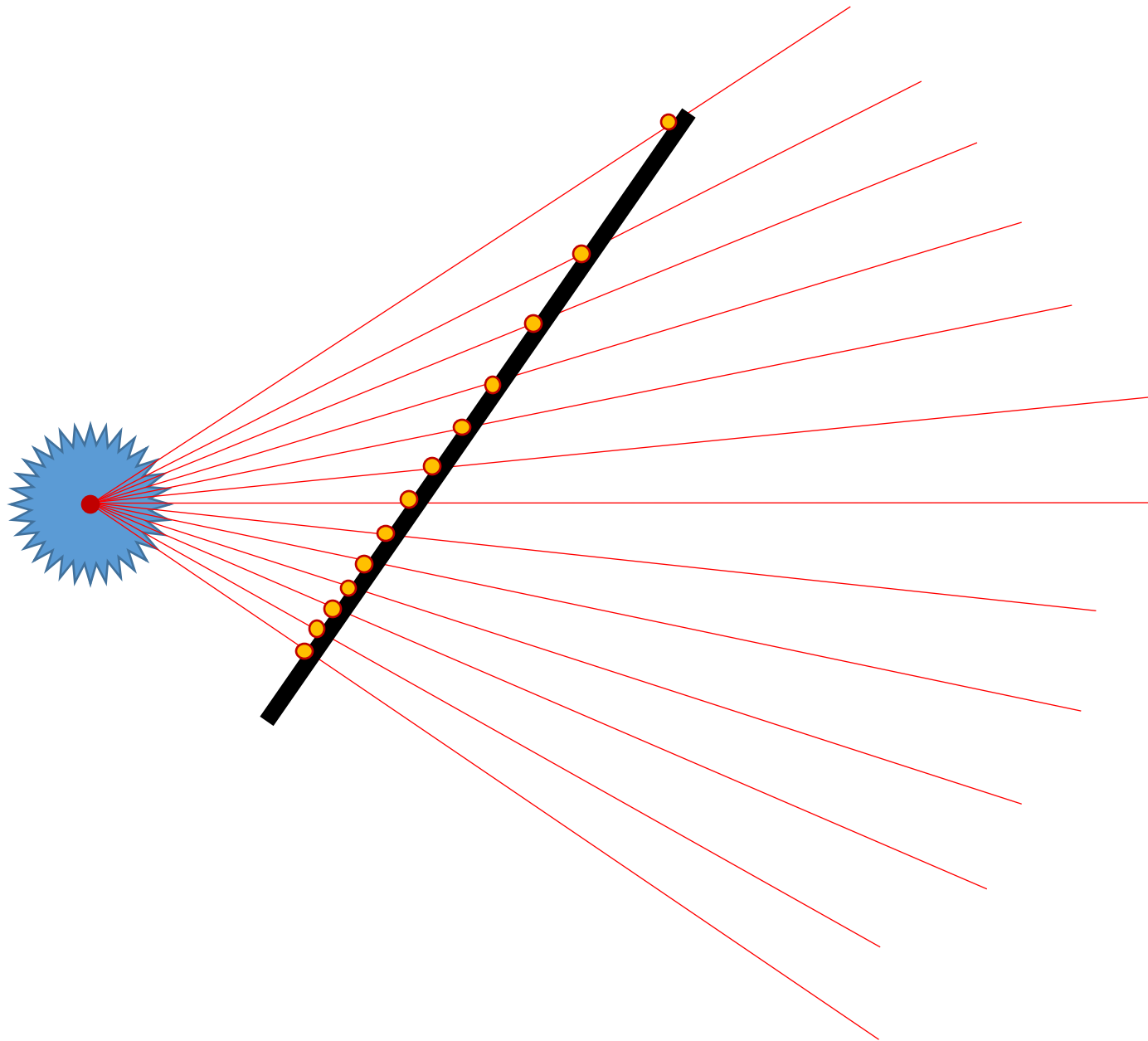


Consider a single long wall.  
The orientation of the wall  
relative to the sensor  
determines the distribution of  
data in the point cloud.

In this example, the data are  
spaced uniformly in space  
(approximately).

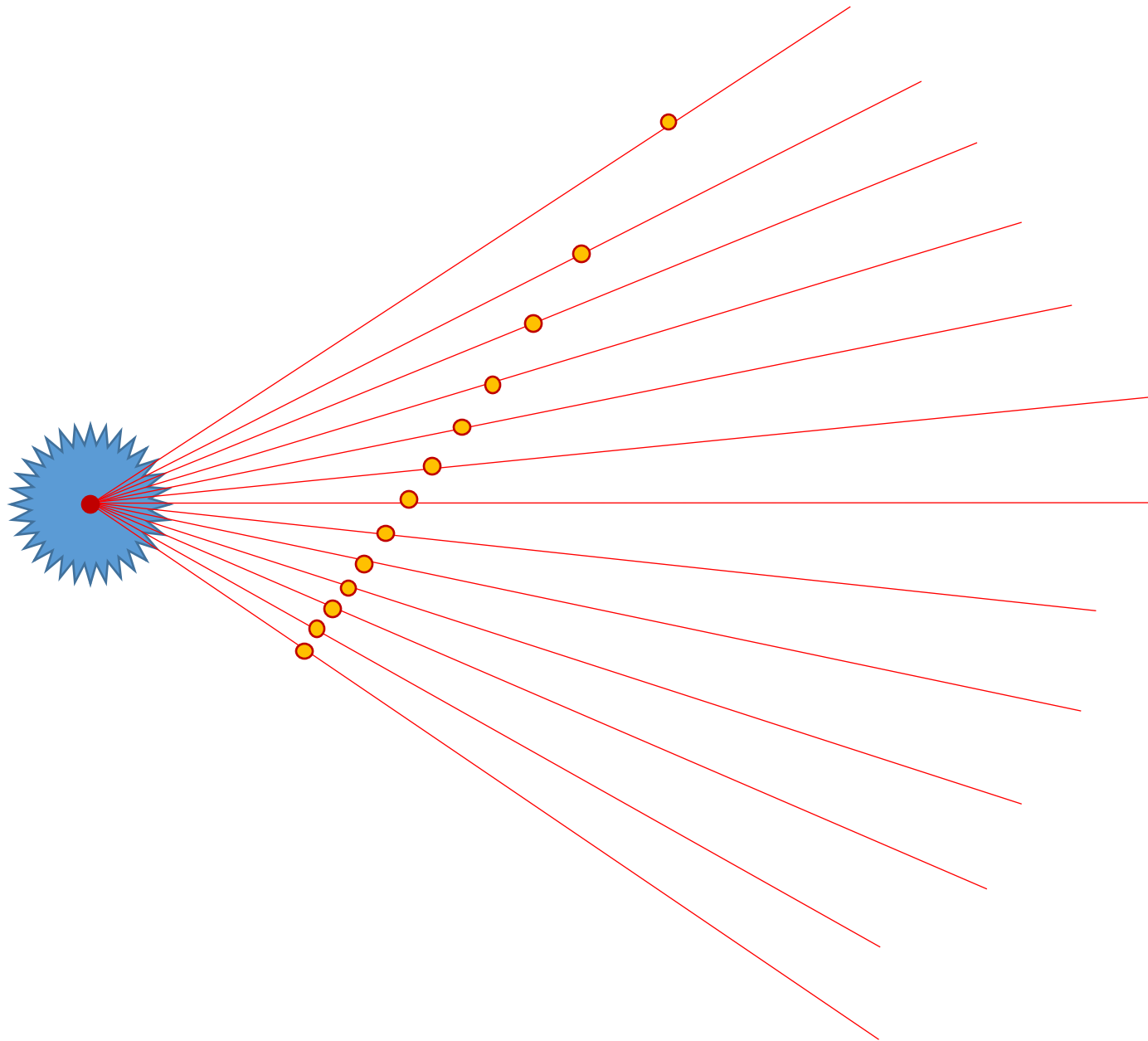


Suppose, however, that the wall is at an oblique angle.



Suppose, however, that the wall is at an oblique angle.

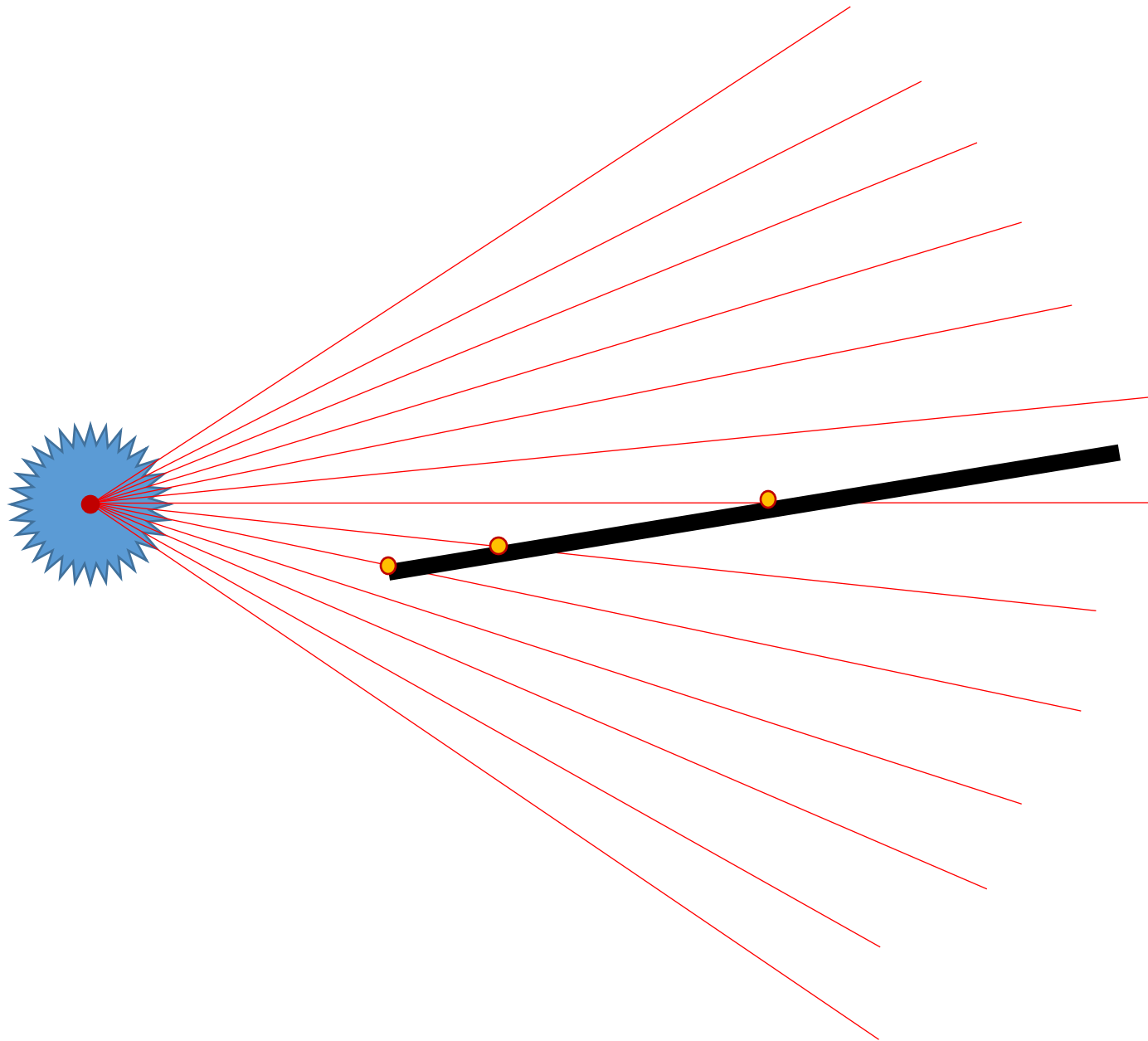




Suppose, however, that the wall is at an oblique angle.

In this case, that data are not at all uniformly spaced.

Sampling uniformity can result in parts of the scene with poor resolution.



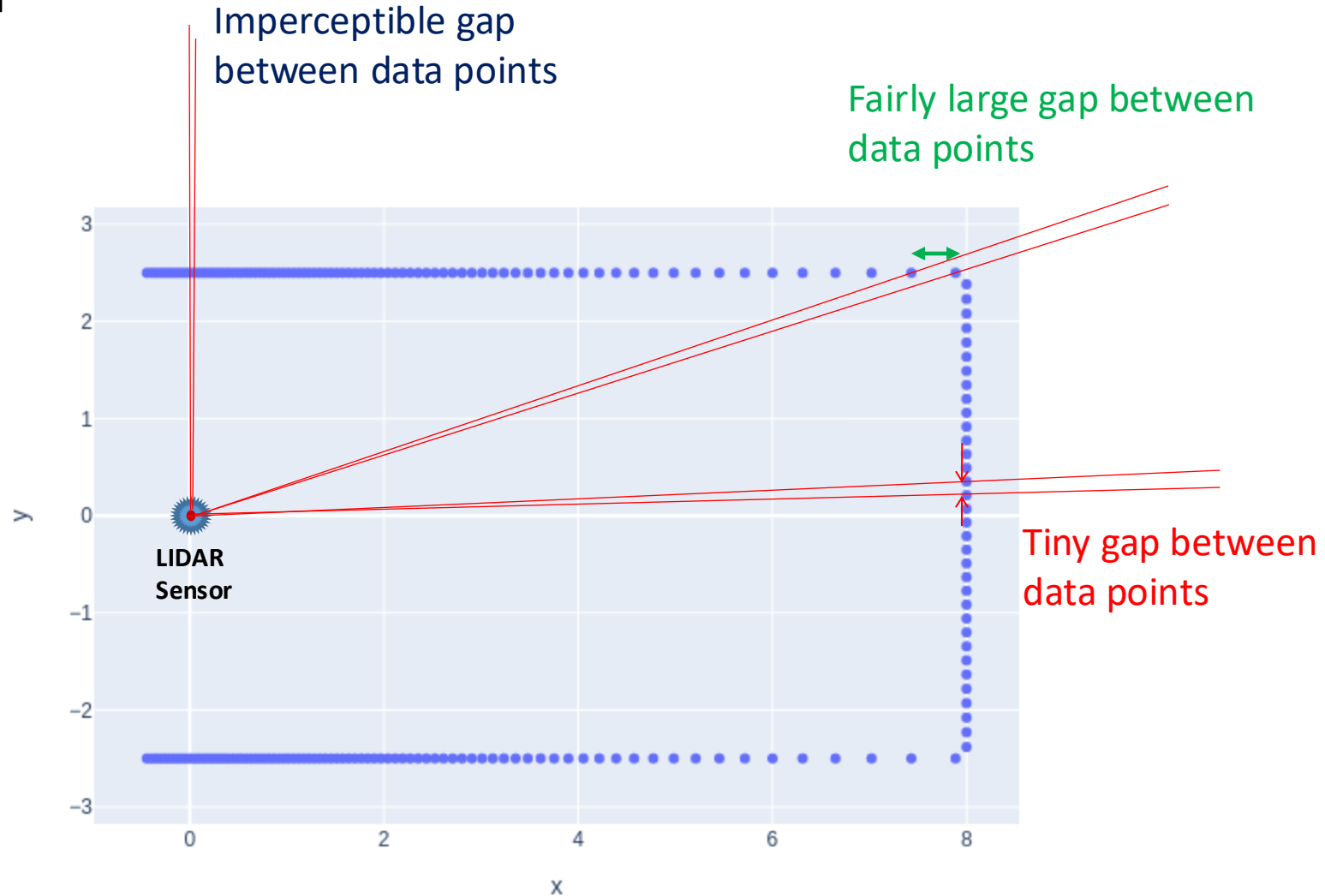
In this case, a fairly large obstacle in the world is represented by only three data points!

You can see these effects in real LIDAR data images.

# Another Example

A robot enters a large, rectangular room, and uses its LIDAR sensor to build a model of the room.

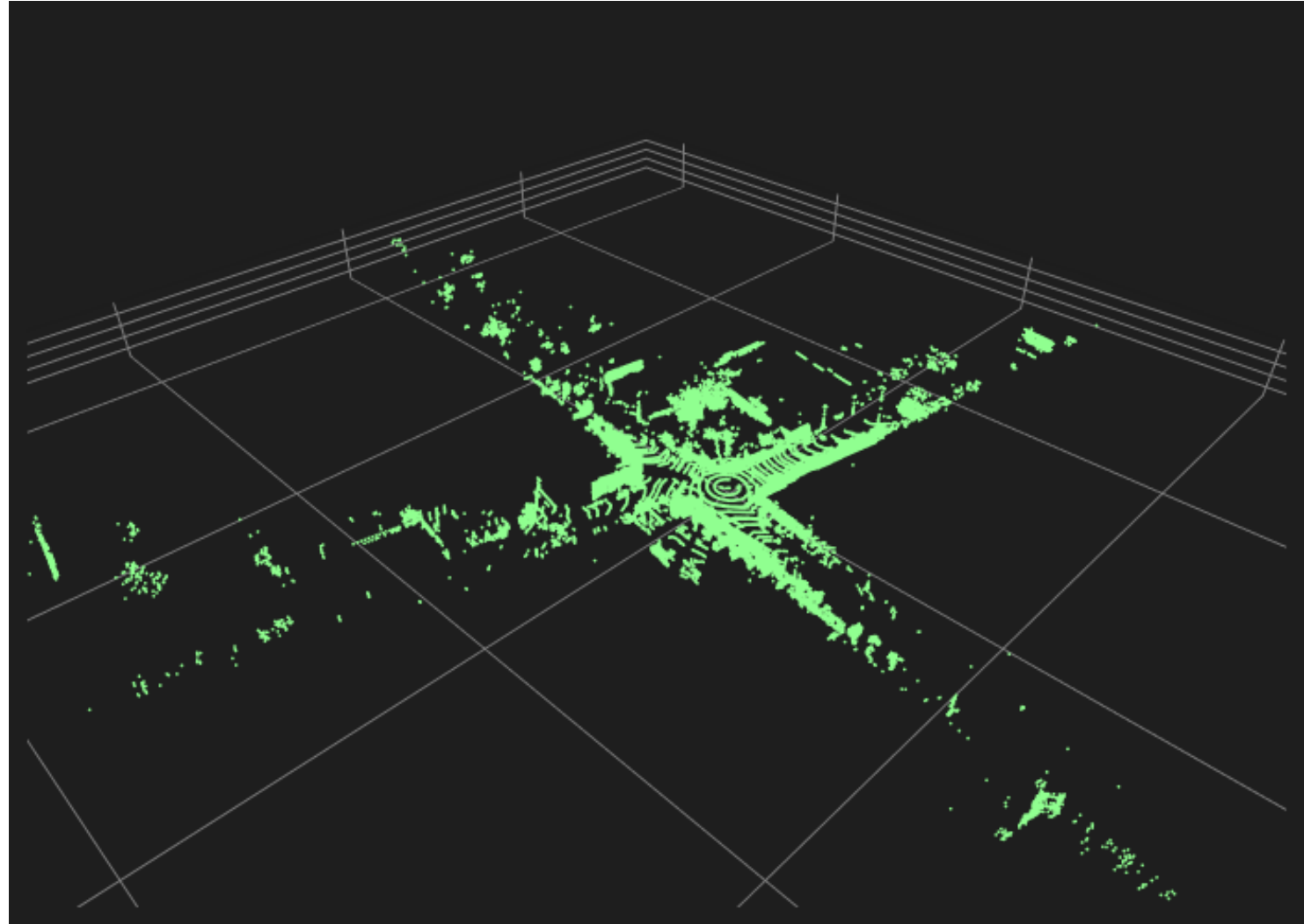
- Points that are nearby and on a surface perpendicular to rays give dense data.
- Points that are distant and on walls that are not parallel to rays give sparse data.
- Points that are distant, but on walls that are perpendicular to rays give moderately dense data.



# Real Data (from the book)

Scan taken when the Argo test car is turning at an intersection:

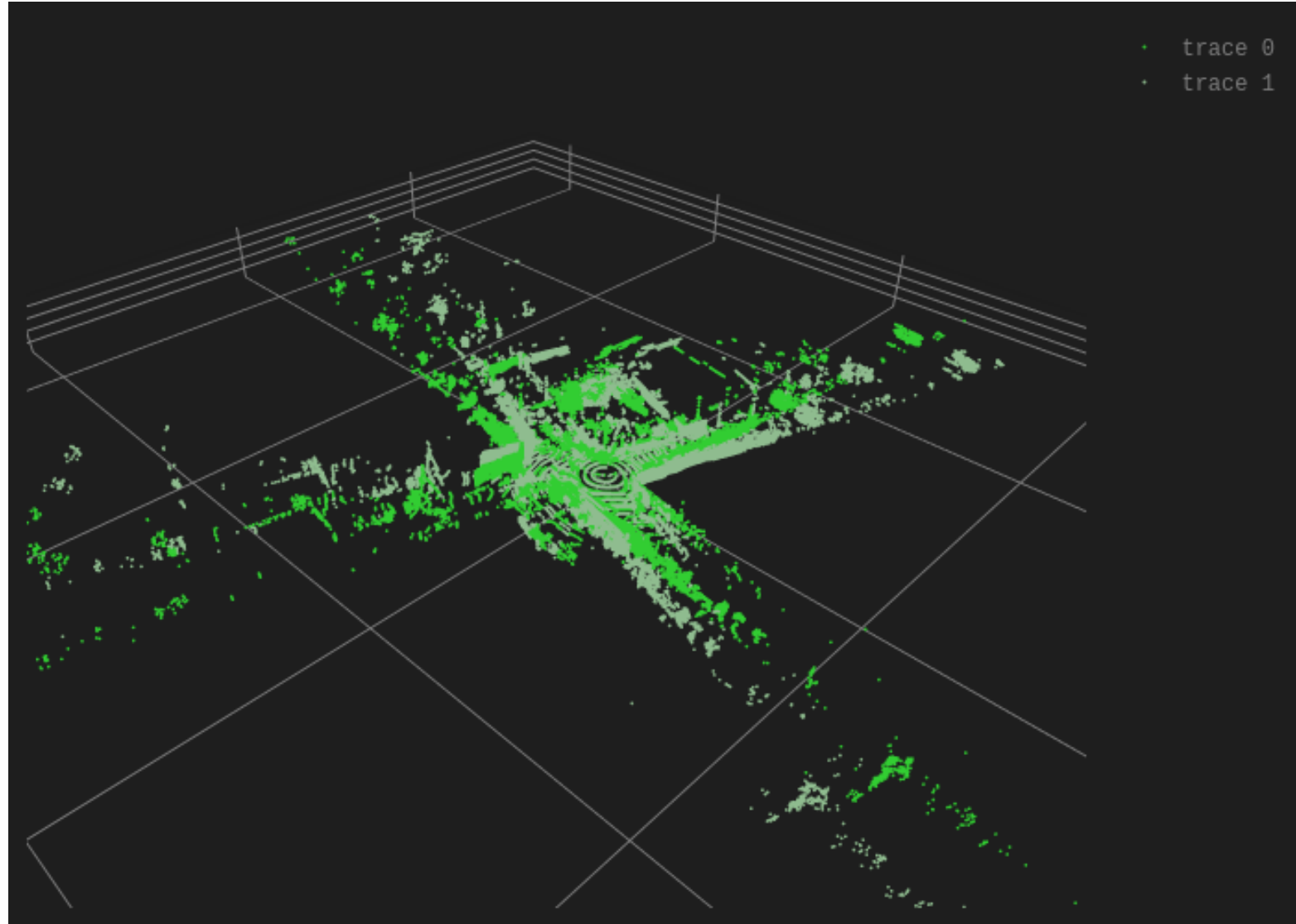
- The location of the car (at the origin) is marked by concentric circles formed by the lowest beams.
- The range is approximately 200m, so we can see fairly far down the cross streets.
- Occlusion is significant: objects close to the car throw “occlusion shadows”.
- Everything is in *body coordinates*, and the fact that the streets appear rotated betrays that the car is actually turning.





# Real Data (from the book)

- This image shows two scans
  - two slightly different colors for the data points.
- Both scans are shown w.r.t. the body coordinate frame of the car.
- These aren't successive scans; eight scans have been omitted.
- Notice the difference in orientation of the data: The car is turning!
- Notice also a slight translation: The car is also moving forward.



# Building a Point Cloud

- As the sensor moves through its environment, it performs a sequence of scans.
- Each scan acquires a set of points in the world, and the coordinates of these points (either  $(x, y)$  for planar data or  $(x, y, z)$  for 3D data) are ***expressed with respect to the coordinate frame of the sensor***.
- To make life easier, we will assume that the body frame of the robot is coincident with the coordinate frame of the sensor.
- In order to build a single map of the environment, it is necessary to express these points in a single coordinate system.
- For most applications, we map points to a single, world coordinate frame.
- This frame could be defined in terms of the local environment (e.g., the origin of the frame could be in Klaus), or in terms of GPS coordinates.
- We will learn how to do this using coordinate transformations!

# Sensing: LIDAR

If we want to do real-time planning in dynamic environments, we need to be able to sense and understand the environment.

- Cameras provide enormous amounts of data, but computer vision remains a difficult problem.
- In general, the computation cost and the error rates for computer vision systems have prevented them from being adopted as the sensor of choice for self-driving cars.
- LIDAR is a fast and effective sensor that can be used to build 3D point clouds in real time.
- LIDAR is the sensor of choice for most robotics systems (including self-driving cars) that navigate in the real world.