# CiviSense

## Geospatial Civic Data in VR

Team ZRD
Zoe Berra
Rudyard Richter
Daniel Parker

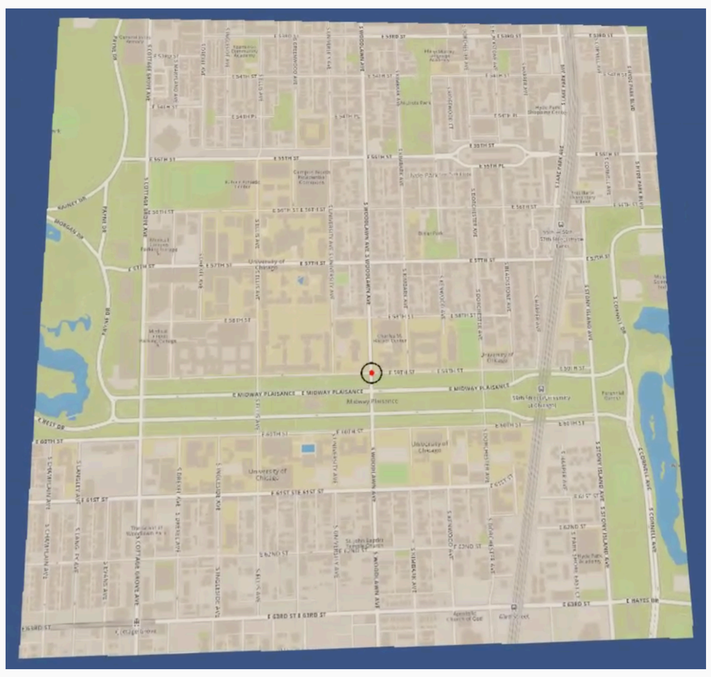
## Objectives

Our goal for CiviSense was twofold: to create an immersive VR experience at any specified location in Chicago; and to augment that experience with representations of location-specific data presented in an intuitive format. To achieve the first goal, we sought to implement a detailed map with simple navigational controls, and render a believable scene at any chosen location. To achieve the second involved selecting which data sets to prioritize in our implementation, as well as presenting these data sets to the user in a natural sensory format.

## Features

The primary feature that CiviSense delivers is a VR experience of a selected location in Chicago, augmented with sensory feedback in visual and auditory forms which represent the frequency of certain geospatial data sets in a given radius around that location.

## Example of Map Interface



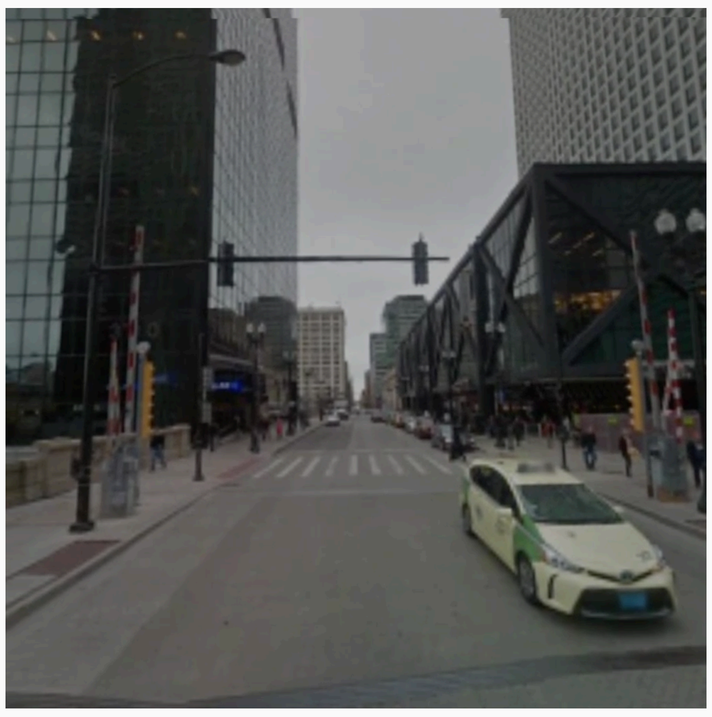
## User Interface

For the user interface in the map view, we chose to implement map navigation through swiping and tapping on the Gear VR trackpad, rather than using body gestures or raycasting to buttons, to keep the game overhead low and the viewframe uncluttered. The only object obscuring the map in the viewframe is a small reticle in the center which allows the user to select a specific location. User interface in the street view is minimal; there is no explicit user interface, only the experience of the various sensory representations of the data at that location. The user can, however, tap on the Gear VR touchpad to bring up the textual overlay which contains written information about the location and associated data.

## Implementation

We implemented CiviSense in the Unity 3D engine, built for the Samsung Gear VR platform. We chose Mapbox Unity, an open source SDK based on OpenStreetMap, to create a slippy tile map that automatically loads the number of tiles necessary to render the map in the view frame, providing the user with an interactive but quickly-loaded view of the map. Using a raycaster from the camera to the plane of map tiles, we triangulate the latitude and longitude of the point that the user selects on the map. Once the latitude and longitude are calculated using the raycaster, we build a skybox using two Google APIs: the latitude and longitude are first sent to the "Snap to Road" feature from the Google Maps API, and then passed to the Street View API to obtain an image for each of the six faces of the skybox. Finally, the latitude and longitude are used to locate and sum relevant data in a fixed radius around the selected point, which are normalized based on a precomputed scale for each data set and then translated into their sensory representations with prefab assets in Unity.

## Example of Street View



## Representation of Data

Data sets are represented to the user in both visual and auditory formats. The volumes of the auditory cues, a police siren and a bicycle bell, are both modulated based on the relative frequency of the data they represent at that location. We included two different forms of visual cues: environmental lighting intensity adjusts based on streetlight outage frequency, while rodent extermination requests control the number of rodent models rendered in the scene. To provide further context, we added a toggleable descriptive panel which loads the street address from Google's reverse geocoding API and fetches the number of incidents calculated in the search radius to provide textual information about the data.

## Data Processing

In addition to designing an intuitive representation of the data sets that we chose to implement, various methods of processing the data obtained from the City of Chicago and Plenario were necessary to facilitate their use within the app. The data were universally constrained to within the range of dates March 1, 2016–February 28, 2017 for consistency between the multiple data sets. For each data set, we preprocessed the data by removing empty rows, invalid data, and features which were not necessary to our implementation. Using the maps provided by Plenario and the City of Chicago Data Portal, we precomputed a reasonable radius and range of expected values based on relative density of the data, which are set as constants for the analysis at runtime and used to normalize the frequencies of the data. This normalization is used to assign the appropriate "intensity" of each data representation in the street view.

## Acknowledgements

This project was completed as part of CS 234/334 Mobile Computing (Winter 2017), taught by Prof. Andrew A Chien with TA support by Gushu Li and Ryan Wu. We gratefully acknowledge the generous support of Samsung in providing Gear VR equipment. Geospatial data are sourced from Plenario and the City of Chicago Data Portal. The map view is constructed using Mapbox and OpenStreetMap. The skyboxes are rendered using Google Street View. The background for this poster is derived from Google map data.