Accident Findings Report

Luke Cadagin

11/24/2021

Report of Grand Rapids Accident Analysis

Load necessary packages:

```
library(tidyverse)
library(sf)
library(osmdata)
library(ggpubr)
library(ggforce)
```

Specify size of all graphs in Knitted Documents:

```
knitr::opts_chunk$set(echo = TRUE, fig.width = 16, fig.height = 8)
```

Upload Grand Rapids Crash Dataset (2008 - 2017):

```
crash_data <- read_csv(here::here("data", "CGR_Crash_Data.csv"))
head(crash_data[1:6])
```

```
## # A tibble: 6 x 6
##       X     Y OBJECTID ROADSOFTID BIKE  CITY
##   <dbl> <dbl>    <dbl>      <dbl> <chr> <chr>
## 1 -85.7  42.9        1    2589528 No    Grand Rapids
## 2 -85.6  42.9        2    2593183 No    Grand Rapids
## 3 -85.7  43.0        3    2582102 No    Grand Rapids
## 4 -85.6  42.9        4    2579820 No    Grand Rapids
## 5 -85.7  43.0        5    2594624 No    Grand Rapids
## 6 -85.7  43.0        6    2599372 No    Grand Rapids
```

Configure features necessary for Grand Rapids map (using openstreetmap api)

```
location_gr <- getbb("Grand Rapids") %>%
    opq()

major_roads_gr <- location_gr %>%
    add_osm_feature(key = "highway", value = c("motorway", "trunk", "primary", "secondary", "tertiary"))
    osmdata_sf()

#minor_roads_gr <- location_gr %>%
    #add_osm_feature(key = "highway", value = c("unclassified", "residential")) %>%
    #osmdata_sf()

water_gr <- location_gr %>%
    add_osm_feature(key = "waterway", value = c("river")) %>%
    osmdata_sf()
```

```
boundary_gr <- location_gr %>%
    add_osm_feature(key = "boundary", value = "administrative") %>%
    add_osm_feature(key = "name", value = "Grand Rapids") %>%
    osmdata_sf()
```

```
## Request failed [429]. Retrying in 1 seconds...
```

```
## Request failed [429]. Retrying in 3.5 seconds...
```

Grand Rapids accidents associated with trains:

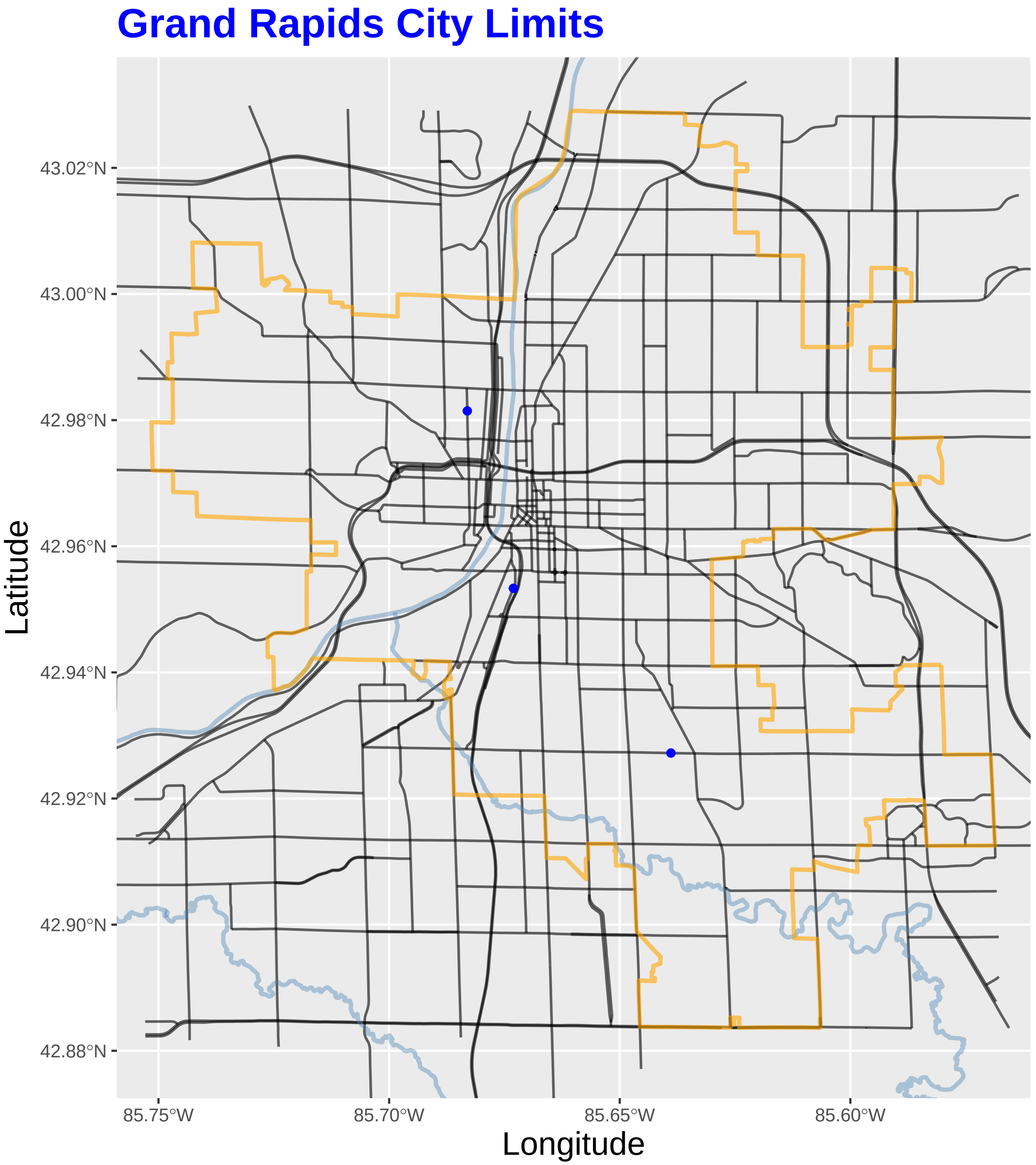
I am interested in studying the impact gates at a railroad crossing have on the number of accidents associated with trains.

Let's start by visualizing the number of Grand Rapids crashes associated with a train from 2008 - 2017.

```
crash_data_train <- crash_data %>%
  filter(TRAIN == "Yes")

crash_data_train %>%
  select(`Longitude` = X, `Latitude` = Y, CRASHDATE, TRAIN, `Principal Road` = PRNAME)
```

```
## # A tibble: 3 x 5
##   Longitude Latitude CRASHDATE  TRAIN `Principal Road`
##       <dbl>    <dbl> <date>     <chr> <chr>
## 1     -85.6     42.9 2008-02-19 Yes   BURTON
## 2     -85.7     43.0 2014-05-03 Yes   11TH
## 3     -85.7     43.0 2017-12-27 Yes   CENTURY
```

```
  ggplot()+
            geom_sf(data = major_roads_gr$osm_lines, size = .6, alpha = .6, color = 'black') +
            #geom_sf(data = minor_roads_gr$osm_lines, size = .3, alpha = .3, color = 'black') +
            geom_sf(data = water_gr$osm_lines, size = 1, alpha = .4, color = 'steelblue') +
            geom_sf(data = boundary_gr$osm_lines, size = 1, alpha = .6, color = "orange") +
            geom_point(data = crash_data_train, mapping = aes(x = X, y = Y), color = "blue") +
            coord_sf(xlim = c(-85.57, -85.75), ylim = c(42.88, 43.03)) +
            labs(title = "Grand Rapids City Limits", x = "Longitude", y = "Latitude") +
            font("title", size = 20, color = "blue", face = "bold") +
            font("x", size = 16) +
            font("y", size = 16)
```



As you can see, from 2008 to 2017 there were only three accidents that occurred in Grand Rapids involving a train (In 2017, 2014, and 2008).

Although we have a very limited number of accidents directly involving a train, this is not a dead-end for our analysis.

Next let's upload a a dataset from Transportation.gov (https://data.transportation.gov/Railroads/Crossing-Inventory-Data-Current/m2f8-22s6) that provides information about every railroad crossing in the USA:

```r
rr_crossing_data <- read_csv(here::here("data", "Crossing_Inventory_Data_-_Current.csv"))
```

```
## Warning: 6574718 parsing failures.
##  row                                         col            expected               actual
## 1224 Number Other MUTCD 1                        1/0/T/F/TRUE/FALSE 2                   '/home/cadag
## 1349 Highway Traffic Signal Interconnection Code 1/0/T/F/TRUE/FALSE 2                   '/home/cadag
## 1349 Highway Traffic Signal Interconnection      1/0/T/F/TRUE/FALSE For Traffic Signals '/home/cadag
## 1349 Highway Traffic Signal Preemption           1/0/T/F/TRUE/FALSE Simultaneous        '/home/cadag
## 1350 Highway Traffic Signal Interconnection Code 1/0/T/F/TRUE/FALSE 2                   '/home/cadag
## .... ........................................... .................. ................... ...........
## See problems(...) for more details.
```

```
head(rr_crossing_data[1:6])
```

```
## # A tibble: 6 x 6
##   `Revision Date` `Reporting Agency Type I~ `Reporting Agency Typ~ `Reason Code`
##   <date>                                <dbl> <chr>                          <dbl>
## 1 1970-01-01                              1 Railroad                          15
## 2 1970-01-01                              1 Railroad                          15
## 3 1970-01-01                              1 Railroad                          15
## 4 1970-01-01                              1 Railroad                          15
## 5 1970-01-01                              1 Railroad                          15
## 6 1970-01-01                              1 Railroad                          15
## # ... with 2 more variables: Reason Description <chr>, Crossing ID <chr>
```

Now we filter this data for only railroad crossing within the Grand Rapids city limits (Note that there are three crossing located in the city center where Latitude = 42.96336 that I removed as it seems like the long/lang for these were placeholders):

```
rr_crossing_data_gr <- rr_crossing_data %>%
  filter(`State Name` == "MICHIGAN", `City Name` == "GRAND RAPIDS", `Intersecting Roadway` == "Yes", La

head(rr_crossing_data_gr[1:6])
```

```
## # A tibble: 6 x 6
##   `Revision Date` `Reporting Agency Type I~ `Reporting Agency Typ~ `Reason Code`
##   <date>                                <dbl> <chr>                          <dbl>
## 1 1991-04-03                              1 Railroad                          16
## 2 1997-12-31                              1 Railroad                          16
## 3 2001-08-31                              1 Railroad                          16
## 4 2010-01-01                              2 State                             16
## 5 2010-01-01                              2 State                             16
## 6 2010-09-01                              1 Railroad                          16
## # ... with 2 more variables: Reason Description <chr>, Crossing ID <chr>
```

Let’s Make Sure that there are no duplicates in the data:

```
nrow(distinct(rr_crossing_data_gr, Latitude))
```

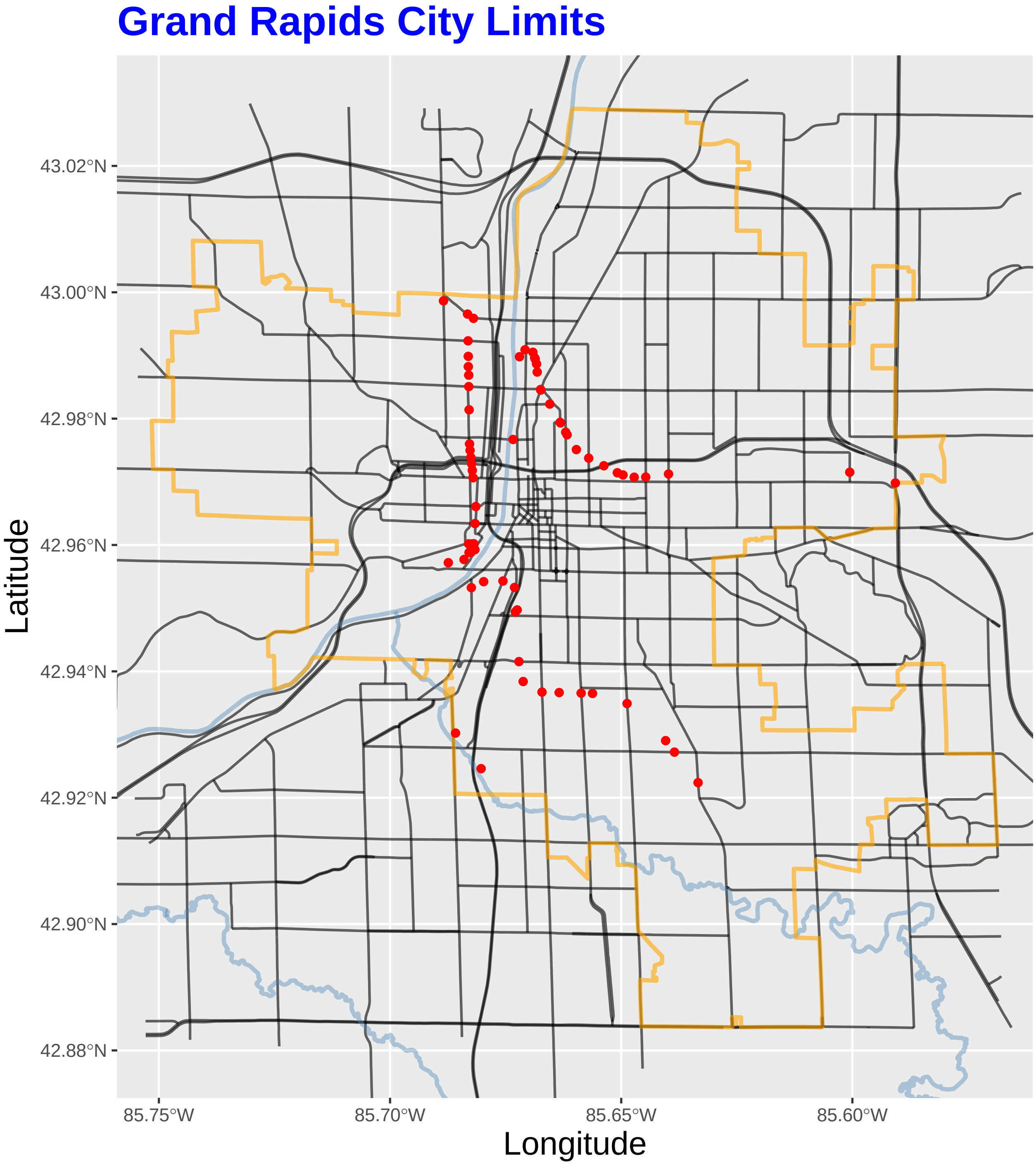
```
## [1] 74
```

```
nrow(rr_crossing_data_gr)
```

```
## [1] 74
```

We can visualize this data in red on our Grand Rapid’s Map:

```
  ggplot()+
            geom_sf(data = major_roads_gr$osm_lines, size = .6, alpha = .6, color = 'black') +
            #geom_sf(data = minor_roads_gr$osm_lines, size = .3, alpha = .3, color = 'black') +
            geom_sf(data = water_gr$osm_lines, size = 1, alpha = .4, color = 'steelblue') +
            geom_sf(data = boundary_gr$osm_lines, size = 1, alpha = .6, color = "orange") +
            geom_point(data = rr_crossing_data_gr, mapping = aes(x = Longitude, y = Latitude), color =
            coord_sf(xlim = c(-85.57, -85.75), ylim = c(42.88, 43.03)) +
            labs(title = "Grand Rapids City Limits", x = "Longitude", y = "Latitude") +
            font("title", size = 20, color = "blue", face = "bold") +
            font("x", size = 16) +
            font("y", size = 16)
```



We now would like to know how many crashes fall within a .0005 (longitudinal units) radius of each railroad crossing.

To do so, we first write a function called in_radius() that detects if a longitude/latitude coordinate is located within a .0005 radius of all railroad crossings:

```
in_radius <- function(x1, y1, x2, y2) {
  if_else(((x1 - x2) ^ 2 + (y1 - y2) ^ 2) <= .0005 ^ 2, 1, 0)
}
```

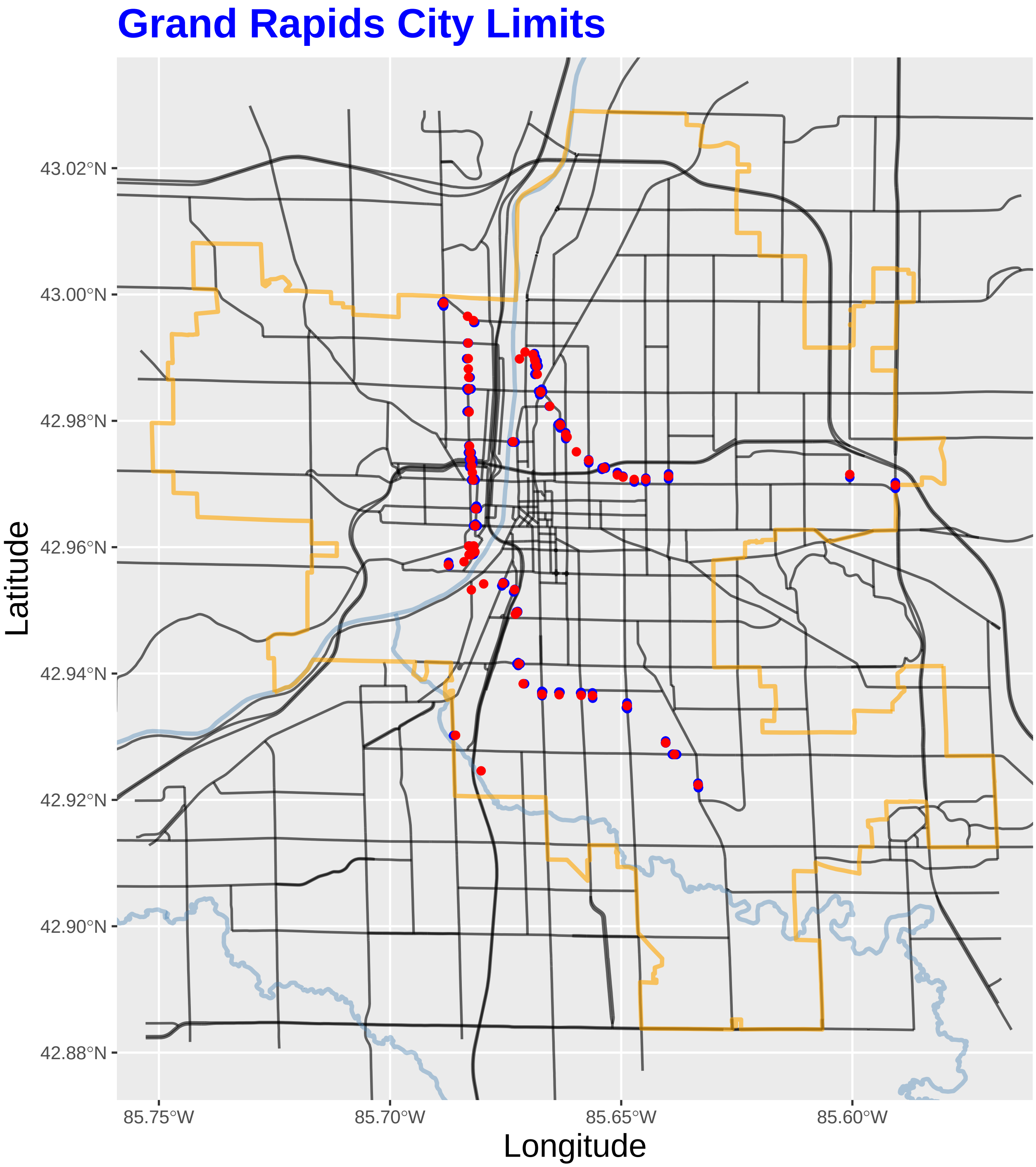
Next we create a new accident subset named near_rr_crash that only contains crashes that occurred within a .0005 radius of a railroad crossing:

```
near_rr_crash <- crash_data %>%
  rowwise() %>%
  filter(1 %in% in_radius(X, Y, rr_crossing_data_gr$Longitude, rr_crossing_data_gr$Latitude))

head(near_rr_crash[1:6])
```

```
## # A tibble: 6 x 6
## # Rowwise:
##       X     Y OBJECTID ROADSOFTID BIKE  CITY
##   <dbl> <dbl>    <dbl>      <dbl> <chr> <chr>
## 1 -85.6  42.9      120    2587376 No    Grand Rapids
## 2 -85.7  43.0      343    2579633 No    Grand Rapids
## 3 -85.6  42.9      663    2591863 No    Grand Rapids
## 4 -85.6  42.9      664    2597546 No    Grand Rapids
## 5 -85.6  42.9      665    2583626 No    Grand Rapids
## 6 -85.6  42.9      666    2579464 No    Grand Rapids
```

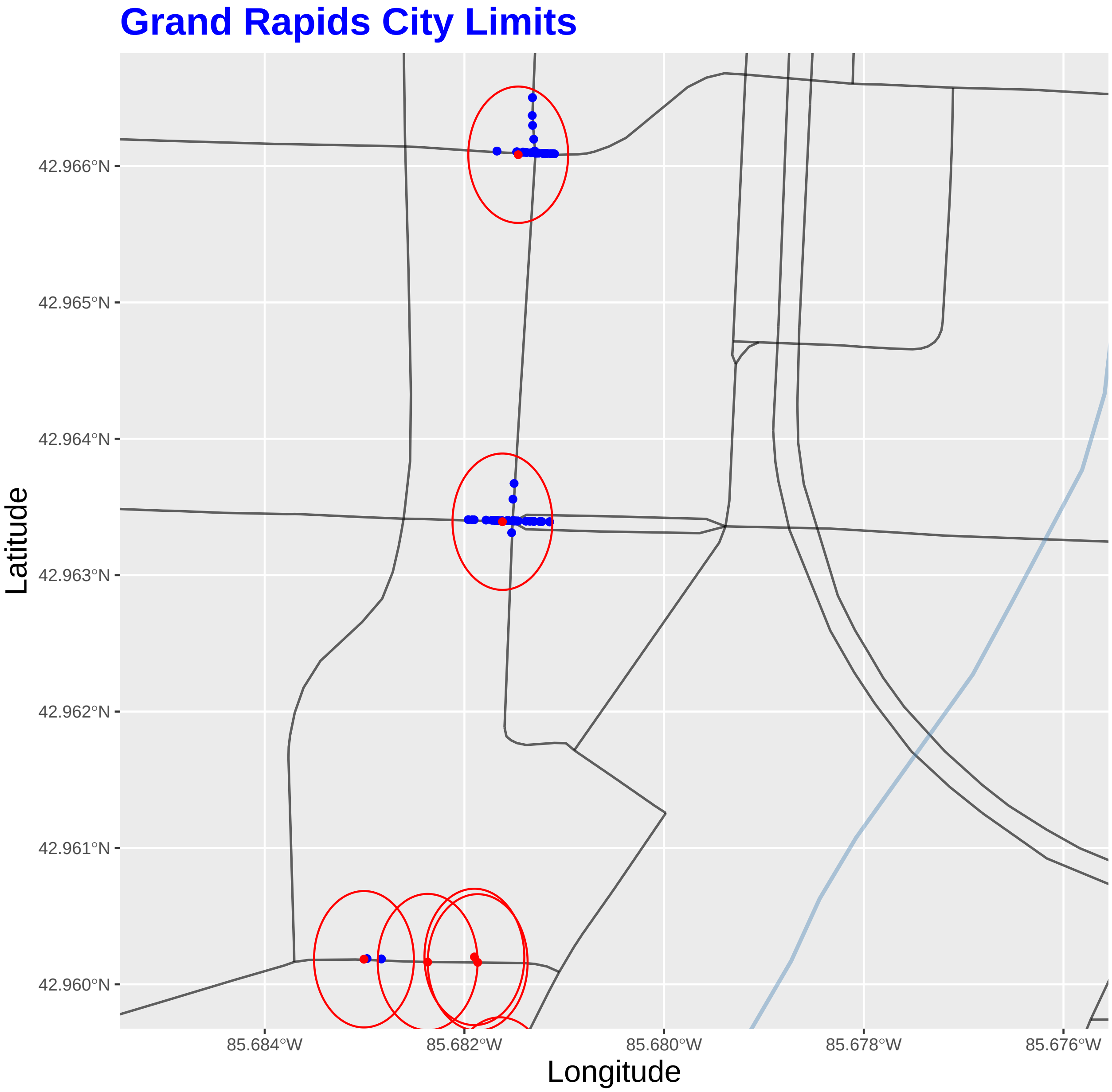
We plot this data on our grand rapids map to visualize the result:

```
  ggplot()+
            geom_sf(data = major_roads_gr$osm_lines, size = .6, alpha = .6, color = 'black') +
            #geom_sf(data = minor_roads_gr$osm_lines, size = .3, alpha = .3, color = 'black') +
            geom_sf(data = water_gr$osm_lines, size = 1, alpha = .4, color = 'steelblue') +
            geom_sf(data = boundary_gr$osm_lines, size = 1, alpha = .6, color = "orange") +
            geom_point(data = near_rr_crash, mapping = aes(x = X, y = Y), color = "blue") +
            geom_point(data = rr_crossing_data_gr, mapping = aes(x = Longitude, y = Latitude), color =
            geom_circle(data = rr_crossing_data_gr, mapping = aes(x0 = Longitude, y0 = Latitude, r = .0
            coord_sf(xlim = c(-85.57, -85.75), ylim = c(42.88, 43.03)) +
            labs(title = "Grand Rapids City Limits", x = "Longitude", y = "Latitude") +
            font("title", size = 20, color = "blue", face = "bold") +
            font("x", size = 16) +
            font("y", size = 16)
```



Below is a zoomed-in portion of the map for better detail (located around the GVSU Pew Campus):

```
ggplot()+
          geom_sf(data = major_roads_gr$osm_lines, size = .6, alpha = .6, color = 'black') +
          #geom_sf(data = minor_roads_gr$osm_lines, size = .3, alpha = .3, color = 'black') +
          geom_sf(data = water_gr$osm_lines, size = 1, alpha = .4, color = 'steelblue') +
          geom_sf(data = boundary_gr$osm_lines, size = 1, alpha = .6, color = "orange") +
          geom_point(data = near_rr_crash, mapping = aes(x = X, y = Y), color = "blue") +
          geom_point(data = rr_crossing_data_gr, mapping = aes(x = Longitude, y = Latitude), color =
          geom_circle(data = rr_crossing_data_gr, mapping = aes(x0 = Longitude, y0 = Latitude, r = .0
          coord_sf(xlim = c(-85.676, -85.685), ylim = c(42.96, 42.9665)) +
          labs(title = "Grand Rapids City Limits", x = "Longitude", y = "Latitude") +
          font("title", size = 20, color = "blue", face = "bold") +
          font("x", size = 16) +
          font("y", size = 16)
```



Now we create a tibble that sums the number of crashes that fall within a .0005 radius of each crossing along with the number of Gate arms at that crossing and the total number of trains that pass through during the day and night:

```
crash_count <- rr_crossing_data_gr %>%
  rowwise() %>%
  mutate(Radius_Count = sum(in_radius(Longitude, Latitude, near_rr_crash$X, near_rr_crash$Y))) %>%
  select(Longitude, Latitude, `Count Roadway Gate Arms`, Radius_Count, `Total Daylight Thru Trains`, `T
  arrange(desc(Radius_Count))
```

```
head(crash_count,10)
```

```
## # A tibble: 10 x 6
## # Rowwise:
##    Longitude Latitude `Count Roadway Gate A~ Radius_Count `Total Daylight Thru ~
##        <dbl>    <dbl>                  <dbl>        <dbl>                  <dbl>
##  1     -85.7     43.0                      0          111                      0
##  2     -85.7     43.0                      0          108                      0
##  3     -85.7     43.0                      0           84                      0
##  4     -85.7     42.9                      2           79                      2
##  5     -85.6     43.0                      0           74                      1
##  6     -85.7     43.0                      0           45                      1
##  7     -85.7     43.0                      0           41                      1
##  8     -85.7     43.0                      2           40                      1
##  9     -85.7     43.0                      0           38                      0
## 10     -85.7     42.9                      2           32                      2
## # ... with 1 more variable: Total Nighttime Thru Trains <dbl>
```

We can use the number of accidents that occur within .0005 longitudinal units from a crossing (Radius_Count) to represent traffic exposure. We are deriving latent information from this variable to determine if additional safety protocols should be put in place at a particular train crossing.

The tibble shows us the crossings with the highest number of crashes that occurred near-by (Radius_Count). We see that only 3 out of 10 of these crossing have gate arms.

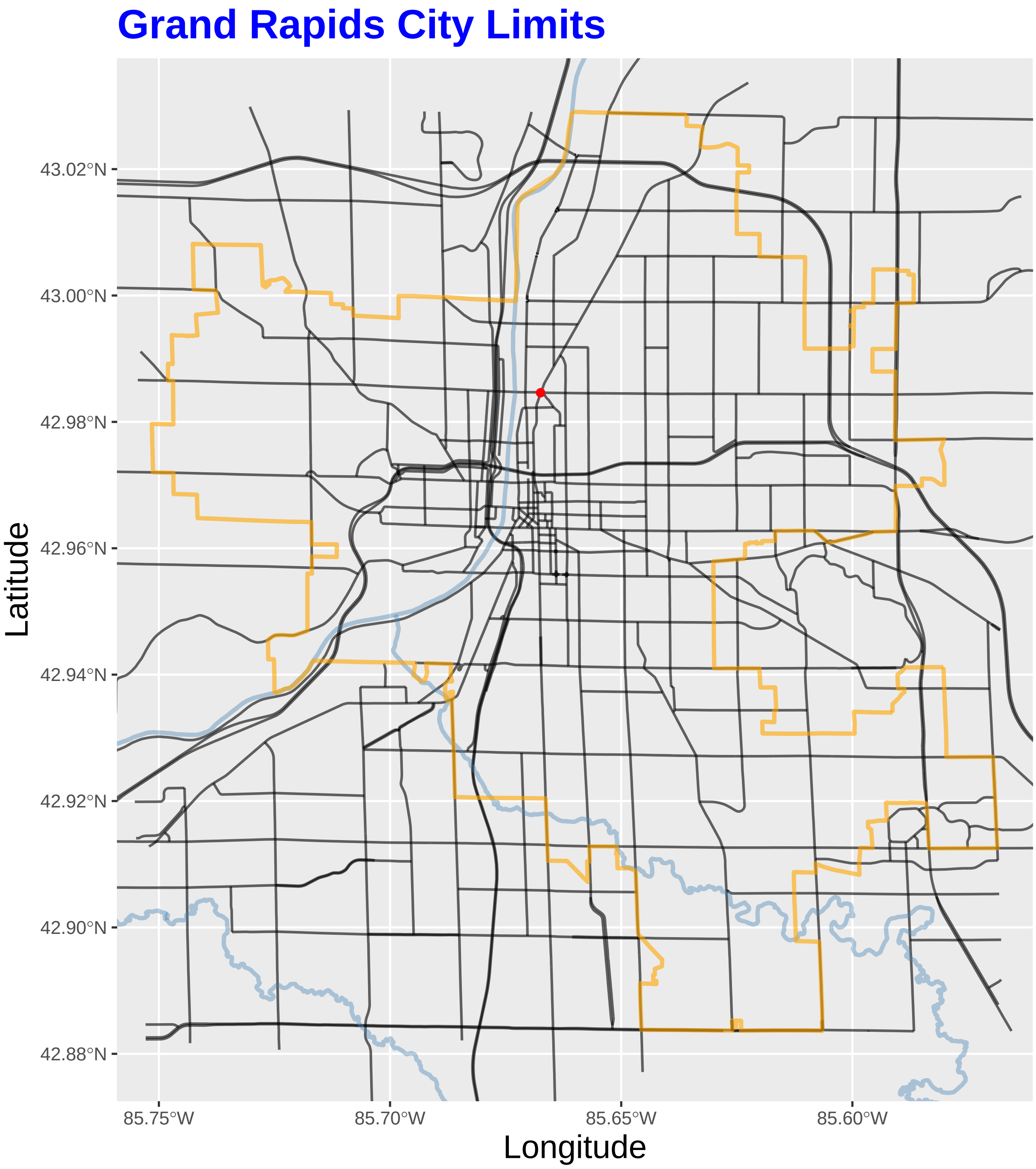
Note, I used the command options(digits = 8) in order to view all of the longitude/latitude digits in the tibbles.

```
test <- rr_crossing_data_gr %>%
  filter(Latitude == 42.984609)
  #filter(Latitude == 42.989799)

test
```

```
## # A tibble: 1 x 248
##   `Revision Date` `Reporting Agency Type I~ `Reporting Agency Typ~ `Reason Code`
##   <date>                              <dbl> <chr>                          <dbl>
## 1 2020-07-10                              1 Railroad                          14
## # ... with 244 more variables: Reason Description <chr>, Crossing ID <chr>,
## #   Crossing ID Suffix <chr>, Reporting Agency Code <chr>,
## #   Reporting Agency Name <chr>, State Code <chr>, State Name <chr>,
## #   County Code <chr>, County Name <chr>, In/Near Code <dbl>, In/Near <chr>,
## #   City Code <chr>, City Name <chr>, City Description <chr>, Street <chr>,
## #   Block Number <lgl>, Highway Name <chr>, Separate Track <chr>,
## #   Separate Track Railroad 1 <chr>, Separate Track Railroad 2 <lgl>,
## #   Separate Track Railroad 3 <lgl>, Separate Track Railroad 4 <lgl>,
## #   Same Track <chr>, Same Track Railroad 1 <chr>, Same Track Railroad 2 <lgl>,
## #   Same Track Railroad 3 <lgl>, Same Track Railroad 4 <lgl>,
## #   Railroad Division <chr>, Railroad Subdivision <chr>, Branch Name <chr>,
## #   Railroad Milepost Prefix <chr>, Railroad Milepost Number <chr>,
## #   Railroad Milepost Suffix <lgl>, Line Segment <chr>,
## #   Nearest Timetable Station <chr>, Timetable Station <dbl>,
## #   Parent Railroad Code <lgl>, Crossing Owner Code <lgl>,
## #   Crossing Type Code <dbl>, Crossing Type <chr>, Crossing Purpose Code <dbl>,
## #   Crossing Purpose <chr>, Crossing Position Code <dbl>,
## #   Crossing Position <chr>, Public Access <lgl>,
## #   Type Of Train Service IDs <lgl>, Type Of Train Service ID 1 <lgl>,
## #   Type Of Train Service 1 <lgl>, Type Of Train Service ID 2 <lgl>,
## #   Type Of Train Service 2 <lgl>, Type Of Train Service ID 3 <lgl>,
## #   Type Of Train Service 3 <lgl>, Type Of Train Service ID 4 <lgl>,
## #   Type Of Train Service 4 <lgl>, Type Of Train Service ID 5 <lgl>,
## #   Type Of Train Service 5 <lgl>, Type Of Train Service ID 6 <lgl>,
## #   Type Of Train Service 6 <lgl>, Less Than One Passenger Train Per Day <lgl>,
## #   Number Passenger Train Per Day <dbl>, Development Type Code <dbl>,
## #   Development Type <chr>, Adjacent Crossing <lgl>,
## #   Adjacent Crossing Number <lgl>, Whistleban Code <dbl>, Whistle Ban <chr>,
## #   Whistle Date <lgl>, High-Speed Rail Corridor ID Suffix <lgl>,
## #   High-Speed Rail Corridor ID <lgl>, Latitude <dbl>, Longitude <dbl>,
## #   Lat/Long Source Code <dbl>, Lat/Long Source <chr>, Railroad Use <lgl>,
## #   Railroad Narrative <lgl>, State Use <lgl>, State Narrative <lgl>,
## #   Emergency Telephone Number <dbl>, Railroad Contact Telephone Number <lgl>,
## #   State Contact Telephone Number <dbl>, Total Daylight Thru Trains <dbl>,
## #   Total Nighttime Thru Trains <dbl>, Total Switching Trains <dbl>,
## #   Total Transit Trains <lgl>, Movements Per Day Code <lgl>,
## #   Movements Per Day <lgl>, Trains Per Week <lgl>,
## #   Trains Per Week Captured Year <lgl>, Maximum Timetable Speed <dbl>,
## #   Typical Minimum Speed Over Crossing <dbl>,
## #   Typical Maximum Speed Over Crossing <dbl>, Number Of Main Tracks <dbl>,
## #   Number Of Siding Tracks <dbl>, Number Of Yard Tracks <lgl>,
## #   Number Of Transit Tracks <lgl>, Number Of Industry Tracks <lgl>,
## #   Train Detection IDs <dbl>, Train Detection ID 1 <dbl>,
## #   Train Detection 1 <chr>, Train Detection ID 2 <lgl>, ...
```

```
ggplot()+
          geom_sf(data = major_roads_gr$osm_lines, size = .6, alpha = .6, color = 'black') +
          #geom_sf(data = minor_roads_gr$osm_lines, size = .3, alpha = .3, color = 'black') +
          geom_sf(data = water_gr$osm_lines, size = 1, alpha = .4, color = 'steelblue') +
          geom_sf(data = boundary_gr$osm_lines, size = 1, alpha = .6, color = "orange") +
          #geom_point(data = near_rr_crash, mapping = aes(x = X, y = Y), color = "blue") +
          geom_point(data = test, mapping = aes(x = Longitude, y = Latitude), color = "red") +
          geom_circle(data = test, mapping = aes(x0 = Longitude, y0 = Latitude, r = .0005), color = "
          coord_sf(xlim = c(-85.57, -85.75), ylim = c(42.88, 43.03)) +
          labs(title = "Grand Rapids City Limits", x = "Longitude", y = "Latitude") +
          font("title", size = 20, color = "blue", face = "bold") +
          font("x", size = 16) +
          font("y", size = 16)
```

```
## Warning in CPL_transform(x, crs, aoi, pipeline, reverse, desired_accuracy, :
## GDAL Error 1: PROJ: proj_as_wkt: DatumEnsemble can only be exported to WKT2:2019

## Warning in CPL_transform(x, crs, aoi, pipeline, reverse, desired_accuracy, :
## GDAL Error 1: PROJ: proj_as_wkt: DatumEnsemble can only be exported to WKT2:2019

## Warning in CPL_transform(x, crs, aoi, pipeline, reverse, desired_accuracy, :
## GDAL Error 1: PROJ: proj_as_wkt: DatumEnsemble can only be exported to WKT2:2019

## Warning in CPL_transform(x, crs, aoi, pipeline, reverse, desired_accuracy, :
## GDAL Error 1: PROJ: proj_as_wkt: DatumEnsemble can only be exported to WKT2:2019

## Warning in CPL_transform(x, crs, aoi, pipeline, reverse, desired_accuracy, :
## GDAL Error 1: PROJ: proj_as_wkt: DatumEnsemble can only be exported to WKT2:2019
## Warning in CPL_transform(x, crs, aoi, pipeline, reverse, desired_accuracy, :
## GDAL Error 1: PROJ: proj_as_wkt: DatumEnsemble can only be exported to WKT2:2019

## Warning in CPL_transform(x, crs, aoi, pipeline, reverse, desired_accuracy, :
## GDAL Error 1: PROJ: proj_as_wkt: DatumEnsemble can only be exported to WKT2:2019

## Warning in CPL_transform(x, crs, aoi, pipeline, reverse, desired_accuracy, :
## GDAL Error 1: PROJ: proj_as_wkt: DatumEnsemble can only be exported to WKT2:2019
```



```
head(crash_data[1:6])
```

```
## # A tibble: 6 x 6
##       X     Y OBJECTID ROADSOFTID BIKE  CITY
##   <dbl> <dbl>    <dbl>      <dbl> <chr> <chr>
## 1 -85.7  42.9        1    2589528 No    Grand Rapids
## 2 -85.6  42.9        2    2593183 No    Grand Rapids
## 3 -85.7  43.0        3    2582102 No    Grand Rapids
## 4 -85.6  42.9        4    2579820 No    Grand Rapids
## 5 -85.7  43.0        5    2594624 No    Grand Rapids
## 6 -85.7  43.0        6    2599372 No    Grand Rapids
```