Weight and Veterans' Environments Study

Road safety features: Weight and Veterans' Environments Study GIS protocol

Xiang W, Jones KK, Matthews SA, Zenk SN.

UIC Neighborhoods + Health

Overview

This protocol describes the process by which Navteq data was used to create road safety measures, in particular measures of low mobility streets, roundabouts, and divided streets.

Acknowledgements

This protocol was developed with grant support from the National Cancer Institute (R21CA195543) and the Department of Veterans Affairs (IIR 13-085), co-led by Shannon Zenk and Elizabeth Tarlov. We thank Sandy Slater for input into this protocol. Haytham Abu Zayd, Shubhayan Ukil and Abby Klemp helped to edit the protocol for dissemination.

Suggested Citation

Xiang W, Jones K, Matthews SA, Zenk SN. (2018). Road safety features: Weight and Veterans' Environments Study GIS protocol, Version 1. Retrieved from Weight and Veterans' Environments Study website: https://waves.uic.edu/.

Table of Contents

Background

This document describes the work process used by WAVES for constructing road safety measures.

Data

Sources

The data source is 2010, 2011, 2013, 2014 Streets file from ArcGIS StreetMap Premium (Navteq source).

Definitions

Road Safety definition and years

The group used these attributes in the Streets dataset to create road safety measures:

- Low mobility streets
- Roundabout
- Streets with dividers
- On-street parking available streets

As an overview, we first create a raster dataset of all eligible streets (excluding non-walkable streets such as tunnels, bridges, highways, etc.). From the eligible streets dataset, we create rasters with subsets of streets that have each of the above features. These input raster datasets are linked to the participant points, and then from the linked result, the percent value can be calculated. For example, the percent of low mobility streets can be calculated as 100*(grid value of low mobility)/ (grid value of all streets).

All road safety measures except On-street Parking are available in year 2010, 2011, 2013 and 2014. On-street Parking is available in year 2010 and 2011.

Decisions

Create road safety measures base raster files

General protocol for creating the road safety measures is:

1. Create eligible "all streets". This excludes streets such as tunnels, bridges, highway, non-paved streets, etc.
2. Use the eligible "all streets", create subsets of streets that have each of these features:
    - Low mobility
    - Roundabout
    - Dividers
    - On-street parking
3. Convert each subset streets (as well as full all streets set) to raster files, via "Polyline to Raster" tool.

4. Create raster files of each subset that identify total amount of each type of street within a given distance around each cell centroid (i.e., a Euclidean buffer around the centroid point), via "Focal Statistics" tool.

Step 1-3 are done in one Python script, and step 4 (grid generation) is done in a separate Python script. The detailed feature selection is additionally documented in Appendix, A.

Step 1-3 Python scripts:

Page 9: 2010&2011

Page 11: 2013

Pgae 14: 2014

Step 4: Generation of buffered rasters

Using the base raster files as input, we used "Focal Statistics" tool to create raster files for 2010, 2011, 2013 and 2014 measuring the total amount of the given types of road features within given distances of each cell centroid:

- Eligible all streets
- Low mobility
- Roundabout
- Dividers
- On-street parking (2010&2011 only)

Distances: 400m, 1600m.

Scripts are:

Page 17: Grid - 2010&2011

Page 19: Grid - 2013&2014

Appendix

A: Scripts

Note: Script will need to be adjusted with project-specific file names and locations.

Note: As field/variable names changed throughout the data years, multiple scripts are included using the current variable names.

Note: **The software used is ArcGIS 10.3.1 and Python 2.7.**

2010 & 2011:

```
#This script is the first step in the creation of Road Safety measures
'''
Step 1. The denominator main Street file is excluded using this selection:
"TUNNEL" = 'Y' OR "BRIDGE" = 'Y' OR "SPEED_CAT" = '8' OR "POIACCESS" = 'Y' OR "FERRY_TYPE" = 'R'
OR "FERRY_TYPE" = 'B' OR "PLOT_ROAD" = 'Y' OR "RAMP" = 'Y' OR "PAVED" = 'N'
OR "INTERINTER" = 'Y' OR "CONTRACC" = 'Y'
Step 2. This script creates these types of % measures, based on Streets file:
"LOW_MBLTY" = '1'
"ROUNDABOUT" = 'Y'
"PARK_AVAIL" = 2 OR "PARK_AVAIL" = 3 OR "PARK_AVAIL" = 4
"DIVIDER" = '1' OR "DIVIDER" = '2' OR "DIVIDER" = 'L' OR "DIVIDER" = 'A'
Step 3. Use Polyline to Raster to convert the above selected lines for each measure to raster

'''
#THIS CAN BE RUN STAND-ALONE IN 64-BIT BACKGROUND GEOPROCESSING IN IDLE
#OTHERWISE IT NEEDS TO BE RUN INSIDE ARCMAP PYTHON WINDOW. DIDN'T WORK IN IPYTHON, "OUT
OF MEMORY" ERROR.
#MAY WORK IN IPYTHON IF IPYTHON IS REFERENCED TO THE 64-BIT BACKGROUD GEOPROCESSING
VERSION


import arcpy, sys, os, time

#set up workspace
in_workspace = {insert file location}
out_workspace = {insert file location}
arcpy.env.extent = "-2493045.0 -1429501.25 2342655.0 1703218.75"
```

```
streets_Name = "streets2011"

arcpy.env.workspace = in_workspace
arcpy.env.overwriteOutput = True

#assign street file to street variable
features = arcpy.ListFeatureClasses()

for feature in features:
                    if feature.find(streets_Name)>-1:
                                streets = feature


#define layer variables

lyr_streets = "lyr_streets"
lyr_streets_intrsctY =  "lyr_streets_intrsctY"

print "Inputs:\n"
print streets
#make layer file for needed feature classes (SelectLayerByAttribute only works on lyr file)
arcpy.MakeFeatureLayer_management(streets, lyr_streets)

start = time.time()

#Step 1. Use Street file from zlevel, convert "Roundabout = Y" to IntrsctElg = Y
fields = arcpy.ListFields(lyr_streets, "count")
if len(fields)!=1:
        arcpy.AddField_management(lyr_streets, "count", "SHORT", "", "", "", "", "NULLABLE",
"NON_REQUIRED")
        arcpy.CalculateField_management(lyr_streets, "count", 1,"PYTHON_9.3")
msg = "Step 1. Converting Roundabout streets to IntrsctElg = Y..."
print msg
where = '"ROUNDABOUT" = \'Y\''
arcpy.SelectLayerByAttribute_management(lyr_streets, "NEW_SELECTION", where)
arcpy.CalculateField_management(lyr_streets, "IntrsctElg", '"Y"',"PYTHON_9.3")

#Step 2. Convert Eligible all streets to raster
msg = "Step 2. Eligible all streets to raster..."
print msg
where = '"IntrsctElg" = \'Y\''
arcpy.SelectLayerByAttribute_management(lyr_streets, "NEW_SELECTION", where)
arcpy.FeatureClassToFeatureClass_conversion(lyr_streets, out_workspace, streets+"_Intrsct_Y")
#convert eligible all streets to raster
arcpy.MakeFeatureLayer_management(out_workspace+streets+"_Intrsct_Y", lyr_streets_intrsctY)
arcpy.PolylineToRaster_conversion(lyr_streets_intrsctY, "count",
out_workspace+streets+"_Intrsct_Y_raster", "MAXIMUM_LENGTH", "", 30)
```

```
#Step 3. Convert LOW_MBLTY streets to raster
msg = "Step 3. LOW_MBLTY streets to raster..."
print msg
where = '"LOW_MBLTY" = \'1\''
arcpy.SelectLayerByAttribute_management(lyr_streets_intrsctY, "NEW_SELECTION", where)
arcpy.PolylineToRaster_conversion(lyr_streets_intrsctY, "count",
out_workspace+streets+"_LOW_MBLTY_raster", "MAXIMUM_LENGTH", "", 30)

#Step 4. Convert "ROUNDABOUT" = 'Y' streets to raster
msg = "Step 4. ROUNDABOUT streets to raster..."
print msg
where = '"ROUNDABOUT" = \'Y\''
arcpy.SelectLayerByAttribute_management(lyr_streets_intrsctY, "NEW_SELECTION", where)
arcpy.PolylineToRaster_conversion(lyr_streets_intrsctY, "count",
out_workspace+streets+"_RDABT_raster", "MAXIMUM_LENGTH", "", 30)

#Step 5. Convert PARKING AVAILABLE streets to raster
msg = "Step 5. PARKING AVAILABLE streets to raster..."
print msg
where = '"PARK_AVAIL" = 2 OR "PARK_AVAIL" = 3 OR "PARK_AVAIL" = 4'
arcpy.SelectLayerByAttribute_management(lyr_streets_intrsctY, "NEW_SELECTION", where)
arcpy.PolylineToRaster_conversion(lyr_streets_intrsctY, "count",
out_workspace+streets+"_PKAVAIL_raster", "MAXIMUM_LENGTH", "", 30)

#Step 6. Convert DIVIDER streets to raster
msg = "Step 6. DIVIDER streets to raster..."
print msg
where = '"DIVIDER" = \'1\' OR "DIVIDER" = \'2\' OR "DIVIDER" = \'L\' OR "DIVIDER" = \'A\''
arcpy.SelectLayerByAttribute_management(lyr_streets_intrsctY, "NEW_SELECTION", where)
arcpy.PolylineToRaster_conversion(lyr_streets_intrsctY, "count",
out_workspace+streets+"_DIVIDER_raster", "MAXIMUM_LENGTH", "", 30)

end = time.time()
print str((end-start)/60)+" minutes"
```

2013:

```
#This is Road Safety measures
'''
Step 1. The denominator main Street file is excluded using this selection:
"TUNNEL" = 'Y' OR "BRIDGE" = 'Y' OR "SPEED_CAT" = '8' OR "POIACCESS" = 'Y' OR "FERRY_TYPE" = 'R'
OR "FERRY_TYPE" = 'B' OR "PLOT_ROAD" = 'Y' OR "RAMP" = 'Y' OR "PAVED" = 'N'
OR "INTERINTER" = 'Y' OR "CONTRACC" = 'Y'
Step 2. This script creates these types of % measures, based on Streets file:
```

"LOW_MBLTY" = '1'
"ROUNDABOUT" = 'Y'
"PARK_AVAIL" = 2 OR "PARK_AVAIL" = 3 OR "PARK_AVAIL" = 4
"DIVIDER" = '1' OR "DIVIDER" = '2' OR "DIVIDER" = 'L' OR "DIVIDER" = 'A'
Step 3. Use Polyline to Raster to convert the above selected lines for each measure to raster

'''
#THIS CAN BE RUN STAND-ALONE IN 64-BIT BACKGROUND GEOPROCESSING IN IDLE
#OTHERWISE IT NEEDS TO BE RUN INSIDE ARCMAP PYTHON WINDOW. DIDN'T WORK IN IPYTHON, "OUT OF MEMORY" ERROR.
#MAY WORK IN IPYTHON IF IPYTHON IS REFERENCED TO THE 64-BIT BACKGROUD GEOPROCESSING VERSION


```
import arcpy, sys, os, time

#set up workspace
in_workspace = {insert file location}
out_workspace = {insert file location}
arcpy.env.extent = "-2493045.0 -1429501.25 2342655.0 1703218.75"


streets_Name = "Street2013"

arcpy.env.workspace = in_workspace
arcpy.env.overwriteOutput = True

#assign street file to street variable
features = arcpy.ListFeatureClasses()

for feature in features:
                if feature.find(streets_Name)>-1:
                                streets = feature


#define layer variables

lyr_streets = "lyr_streets"
lyr_streets_intrsctY =  "lyr_streets_intrsctY"

print "Inputs:\n"
print streets
#make layer file for needed feature classes (SelectLayerByAttribute only works on lyr file)
arcpy.MakeFeatureLayer_management(streets, lyr_streets)

start = time.time()

#Step 1. Use Street file from zlevel, convert "Roundabout = Y" to IntrsctElg = Y
```

```
fields = arcpy.ListFields(lyr_streets, "count")
if len(fields)!=1:
        arcpy.AddField_management(lyr_streets, "count", "SHORT", "", "", "", "", "NULLABLE",
"NON_REQUIRED")
        arcpy.CalculateField_management(lyr_streets, "count", 1,"PYTHON_9.3")
msg = "Step 1. Converting Roundabout streets to IntrsctElg = Y..."
print msg
where = '"Roundabout" = \'Y\''
arcpy.SelectLayerByAttribute_management(lyr_streets, "NEW_SELECTION", where)
arcpy.CalculateField_management(lyr_streets, "IntrsctElg", '"Y"',"PYTHON_9.3")

#Step 2. Convert Eligible all streets to raster
msg = "Step 2. Eligible all streets to raster..."
print msg
where = '"IntrsctElg" = \'Y\''
arcpy.SelectLayerByAttribute_management(lyr_streets, "NEW_SELECTION", where)
arcpy.FeatureClassToFeatureClass_conversion(lyr_streets, out_workspace, streets+"_Intrsct_Y")
#convert eligible all streets to raster
arcpy.MakeFeatureLayer_management(out_workspace+streets+"_Intrsct_Y", lyr_streets_intrsctY)
arcpy.PolylineToRaster_conversion(lyr_streets_intrsctY, "count",
out_workspace+streets+"_Intrsct_Y_raster", "MAXIMUM_LENGTH", "", 30)



#Step 3. Convert LOW_MBLTY streets to raster
msg = "Step 3. LOW_MBLTY streets to raster..."
print msg
where = '"LOW_MOBILITY" = 1'
arcpy.SelectLayerByAttribute_management(lyr_streets_intrsctY, "NEW_SELECTION", where)
arcpy.PolylineToRaster_conversion(lyr_streets_intrsctY, "count",
out_workspace+streets+"_LOW_MBLTY_raster", "MAXIMUM_LENGTH", "", 30)

#Step 4. Convert "ROUNDABOUT" = 'Y' streets to raster
msg = "Step 4. ROUNDABOUT streets to raster..."
print msg
where = '"Roundabout" = \'Y\''
arcpy.SelectLayerByAttribute_management(lyr_streets_intrsctY, "NEW_SELECTION", where)
arcpy.PolylineToRaster_conversion(lyr_streets_intrsctY, "count",
out_workspace+streets+"_RDABT_raster", "MAXIMUM_LENGTH", "", 30)
'''
# PARKING AVAILABILITY IS NOT IN 2013
#Step 5. Convert PARKING AVAILABLE streets to raster
msg = "Step 5. PARKING AVAILABLE streets to raster..."
print msg
where = '"PARK_AVAIL" = 2 OR "PARK_AVAIL" = 3 OR "PARK_AVAIL" = 4'
arcpy.SelectLayerByAttribute_management(lyr_streets_intrsctY, "NEW_SELECTION", where)
```

```
arcpy.PolylineToRaster_conversion(lyr_streets_intrsctY, "count",
out_workspace+streets+"_PKAVAIL_raster", "MAXIMUM_LENGTH", "", 30)
'''
#Step 6. Convert DIVIDER streets to raster
msg = "Step 6. DIVIDER streets to raster..."
print msg
where = '"DIVIDER" = \'1\' OR "DIVIDER" = \'2\' OR "DIVIDER" = \'L\' OR "DIVIDER" = \'A\''
arcpy.SelectLayerByAttribute_management(lyr_streets_intrsctY, "NEW_SELECTION", where)
arcpy.PolylineToRaster_conversion(lyr_streets_intrsctY, "count",
out_workspace+streets+"_DIVIDER_raster", "MAXIMUM_LENGTH", "", 30)

end = time.time()
print str((end-start)/60)+" minutes"
```

2014:

```
#This is Road Safety measures
'''
Step 1. The denominator main Street file is excluded using this selection:
"TUNNEL" = 'Y' OR "BRIDGE" = 'Y' OR "SPEED_CAT" = '8' OR "POIACCESS" = 'Y' OR "FERRY_TYPE" = 'R'
OR "FERRY_TYPE" = 'B' OR "PLOT_ROAD" = 'Y' OR "RAMP" = 'Y' OR "PAVED" = 'N'
OR "INTERINTER" = 'Y' OR "CONTRACC" = 'Y'
Step 2. This script creates these types of % measures, based on Streets file:
"LOW_MBLTY" = '1'
"ROUNDABOUT" = 'Y'
"PARK_AVAIL" = 2 OR "PARK_AVAIL" = 3 OR "PARK_AVAIL" = 4
"DIVIDER" = '1' OR "DIVIDER" = '2' OR "DIVIDER" = 'L' OR "DIVIDER" = 'A'
Step 3. Use Polyline to Raster to convert the above selected lines for each measure to raster

'''
#THIS CAN BE RUN STAND-ALONE IN 64-BIT BACKGROUND GEOPROCESSING IN IDLE
#OTHERWISE IT NEEDS TO BE RUN INSIDE ARCMAP PYTHON WINDOW. DIDN'T WORK IN IPYTHON, "OUT
OF MEMORY" ERROR.
#MAY WORK IN IPYTHON IF IPYTHON IS REFERENCED TO THE 64-BIT BACKGROUD GEOPROCESSING
VERSION


import arcpy, sys, os, time

#set up workspace
in_workspace = {insert file location}
out_workspace = {insert file location}
arcpy.env.extent = "-2493045.0 -1429501.25 2342655.0 1703218.75"


streets_Name = "Streets2014"
```

```
arcpy.env.workspace = in_workspace
arcpy.env.overwriteOutput = True

#assign street file to street variable
features = arcpy.ListFeatureClasses()

for feature in features:
                    if feature.find(streets_Name)>-1:
                                streets = feature


#define layer variables

lyr_streets = "lyr_streets"
lyr_streets_intrsctY =  "lyr_streets_intrsctY"

print "Inputs:\n"
print streets
#make layer file for needed feature classes (SelectLayerByAttribute only works on lyr file)
arcpy.MakeFeatureLayer_management(streets, lyr_streets)

start = time.time()

#Step 0. Add field "PLOT_ROAD" to streets file
fields = arcpy.ListFields(lyr_streets, "count")
if len(fields)!=1:
        arcpy.AddField_management(lyr_streets, "count", "SHORT", "", "", "", "", "NULLABLE",
"NON_REQUIRED")
        arcpy.CalculateField_management(lyr_streets, "count", 1,"PYTHON_9.3")
arcpy.AddField_management(lyr_streets, "PLOT_ROAD", "TEXT", "", "", "", "", "NULLABLE",
"NON_REQUIRED")
msg = "Step 0. Calculating streets PLOT_ROAD attribute..."
print msg
where = '"FuncClass" = 5 AND "SpeedCat" = 8 AND "LANE_CATEGORY" = 1 AND "RestrictCars" = \'N\'
AND "RestrictPedestrians" = \'N\' AND "RestrictThroughTraffic" = \'Y\''
arcpy.SelectLayerByAttribute_management(lyr_streets, "NEW_SELECTION", where)
arcpy.CalculateField_management(lyr_streets, "PLOT_ROAD", '"Y"',"PYTHON_9.3")
arcpy.SelectLayerByAttribute_management(lyr_streets, "CLEAR_SELECTION")

#Step 1. Add field "IntrsctElg" to street
arcpy.AddField_management(lyr_streets, "IntrsctElg", "TEXT", "", "", "", "", "NULLABLE",
"NON_REQUIRED")
#Calculate unwanted streets as IntrsctElg = N
msg = "Step 1. Calculating unwanted streets as IntrsctElg = N..."
print msg
```

```
where = '"TUNNEL" = \'Y\' OR "BRIDGE" = \'Y\' OR "SpeedCat" = 8 OR "POI_ACCESS" = \'Y\' OR
"FerryType" = \'R\' OR "FerryType" = \'B\' OR "PLOT_ROAD" = \'Y\' OR "Ramp" = \'Y\' OR "Paved" = \'N\'
OR "INTERSECTION_CATEGORY" = \'Y\' OR "ContrAcc" = \'Y\''
arcpy.SelectLayerByAttribute_management(lyr_streets, "NEW_SELECTION", where)
arcpy.CalculateField_management(lyr_streets, "IntrsctElg", '"N"',"PYTHON_9.3")
arcpy.SelectLayerByAttribute_management(lyr_streets, "SWITCH_SELECTION")
arcpy.CalculateField_management(lyr_streets, "IntrsctElg", '"Y"',"PYTHON_9.3")


#Step 2. Convert Eligible all streets to raster
msg = "Step 2. Eligible all streets to raster..."
print msg
where = '"IntrsctElg" = \'Y\''
arcpy.SelectLayerByAttribute_management(lyr_streets, "NEW_SELECTION", where)
arcpy.FeatureClassToFeatureClass_conversion(lyr_streets, out_workspace, streets+"_Intrsct_Y")
#convert eligible all streets to raster
arcpy.MakeFeatureLayer_management(out_workspace+streets+"_Intrsct_Y", lyr_streets_intrsctY)
arcpy.PolylineToRaster_conversion(lyr_streets_intrsctY, "count",
out_workspace+streets+"_Intrsct_Y_raster", "MAXIMUM_LENGTH", "", 30)


#Step 3. Convert LOW_MBLTY streets to raster
msg = "Step 3. LOW_MBLTY streets to raster..."
print msg
where = '"LOW_MOBILITY" = 1'
arcpy.SelectLayerByAttribute_management(lyr_streets_intrsctY, "NEW_SELECTION", where)
arcpy.PolylineToRaster_conversion(lyr_streets_intrsctY, "count",
out_workspace+streets+"_LOW_MBLTY_raster", "MAXIMUM_LENGTH", "", 30)

#Step 4. Convert "ROUNDABOUT" = 'Y' streets to raster
msg = "Step 4. ROUNDABOUT streets to raster..."
print msg
where = '"Roundabout" = \'Y\''
arcpy.SelectLayerByAttribute_management(lyr_streets_intrsctY, "NEW_SELECTION", where)
arcpy.PolylineToRaster_conversion(lyr_streets_intrsctY, "count",
out_workspace+streets+"_RDABT_raster", "MAXIMUM_LENGTH", "", 30)
'''
# PARKING AVAILABILITY IS NOT IN 2013, 2014
#Step 5. Convert PARKING AVAILABLE streets to raster
msg = "Step 5. PARKING AVAILABLE streets to raster..."
print msg
where = '"PARK_AVAIL" = 2 OR "PARK_AVAIL" = 3 OR "PARK_AVAIL" = 4'
arcpy.SelectLayerByAttribute_management(lyr_streets_intrsctY, "NEW_SELECTION", where)
arcpy.PolylineToRaster_conversion(lyr_streets_intrsctY, "count",
out_workspace+streets+"_PKAVAIL_raster", "MAXIMUM_LENGTH", "", 30)
'''
#Step 6. Convert DIVIDER streets to raster
msg = "Step 6. DIVIDER streets to raster..."
```

```
print msg
where = '"DIVIDER" = \'1\' OR "DIVIDER" = \'2\' OR "DIVIDER" = \'L\' OR "DIVIDER" = \'A\''
arcpy.SelectLayerByAttribute_management(lyr_streets_intrsctY, "NEW_SELECTION", where)
arcpy.PolylineToRaster_conversion(lyr_streets_intrsctY, "count",
out_workspace+streets+"_DIVIDER_raster", "MAXIMUM_LENGTH", "", 30)

end = time.time()
print str((end-start)/60)+" minutes"
```

Grid- 2010 &2011:

```
#road saftey grids generation 2010-2011

import arcpy, time
from arcpy.sa import *

in_workspace = {insert file location}
out_workspace = {insert file location}
arcpy.env.workspace = in_workspace
arcpy.env.extent = "-2493045.0 -1429501.25 2342655.0 1703218.75"
arcpy.env.overwriteOutput = True
arcpy.CheckOutExtension("Spatial")

start = time.time()

#set focal statistics variables

neighborhood_400 = NbrCircle(400, "MAP")
neighborhood_1600 = NbrCircle(1600, "MAP")
neighborhood_4800 = NbrCircle(4800, "MAP")
neighborhood_8000 = NbrCircle(8000, "MAP")

# all streets
# Set focal statistics variables
lyr_all = "all_streets_2011"
arcpy.MakeRasterLayer_management("Street2011_Intrsct_Y_raster", lyr_all)
inRaster = lyr_all

# Execute FocalStatistics

outFocalStatistics = FocalStatistics(inRaster, neighborhood_400, "SUM","")
outFocalStatistics.save(out_workspace+"Rdsfty_2011_ALL_400m")
print inRaster
outFocalStatistics = FocalStatistics(inRaster, neighborhood_1600, "SUM","")
outFocalStatistics.save(out_workspace+"Rdsfty_2011_ALL_1600m")
```

```
print inRaster

# Low Mobility Streets
# Set focal statistics variables
lyr_lowmob = "lowmob_2011"
arcpy.MakeRasterLayer_management("Street2011_LOW_MBLTY_raster", lyr_lowmob)
inRaster = lyr_lowmob

# Execute FocalStatistics

outFocalStatistics = FocalStatistics(inRaster, neighborhood_400, "SUM","")
outFocalStatistics.save(out_workspace+"Rdsfty_2011_LOWMOB_400m")
print inRaster
outFocalStatistics = FocalStatistics(inRaster, neighborhood_1600, "SUM","")
outFocalStatistics.save(out_workspace+"Rdsfty_2011_LOWMOB_1600m")
print inRaster


# Parking available Streets
# Set focal statistics variables
lyr_parkavail = "pkavail_2011"
arcpy.MakeRasterLayer_management("Street2011_PKAVAIL_raster", lyr_parkavail)
inRaster = lyr_parkavail

# Execute FocalStatistics

outFocalStatistics = FocalStatistics(inRaster, neighborhood_400, "SUM","")
outFocalStatistics.save(out_workspace+"Rdsfty_2011_PKAVAIL_400m")
print inRaster
outFocalStatistics = FocalStatistics(inRaster, neighborhood_1600, "SUM","")
outFocalStatistics.save(out_workspace+"Rdsfty_2011_PKAVAIL_1600m")
print inRaster

# Roundabout Streets
# Set focal statistics variables
lyr_rdabout = "rdabout_2011"
arcpy.MakeRasterLayer_management("Street2011_RDABT_raster", lyr_rdabout)
inRaster = lyr_rdabout

# Execute FocalStatistics

outFocalStatistics = FocalStatistics(inRaster, neighborhood_400, "SUM","")
outFocalStatistics.save(out_workspace+"Rdsfty_2011_RDABT_400m")
print inRaster
outFocalStatistics = FocalStatistics(inRaster, neighborhood_1600, "SUM","")
outFocalStatistics.save(out_workspace+"Rdsfty_2011_RDABT_1600m")
print inRaster
```

```
# Divider Streets
# Set focal statistics variables
lyr_divider = "divider_2011"
arcpy.MakeRasterLayer_management("Street2011_DIVIDER_raster", lyr_divider)
inRaster = lyr_divider

# Execute FocalStatistics

outFocalStatistics = FocalStatistics(inRaster, neighborhood_400, "SUM","")
outFocalStatistics.save(out_workspace+"Rdsfty_2011_DIVIDER_400m")
print inRaster
outFocalStatistics = FocalStatistics(inRaster, neighborhood_1600, "SUM","")
outFocalStatistics.save(out_workspace+"Rdsfty_2011_DIVIDER_1600m")
print inRaster

end = time.time()
print str((end-start)/60)+" minutes"
```

Grid- 2013 & 2014:

```
#road saftey grids generation 2010-2014

import arcpy, time
from arcpy.sa import *

in_workspace = {insert file location}
out_workspace = {insert file location}
arcpy.env.workspace = in_workspace
arcpy.env.extent = "-2493045.0 -1429501.25 2342655.0 1703218.75"
arcpy.env.overwriteOutput = True
arcpy.CheckOutExtension("Spatial")

start = time.time()

#set focal statistics variables

neighborhood_400 = NbrCircle(400, "MAP")
neighborhood_1600 = NbrCircle(1600, "MAP")
neighborhood_4800 = NbrCircle(4800, "MAP")
neighborhood_8000 = NbrCircle(8000, "MAP")

# all streets
# Set focal statistics variables
lyr_all = "all_streets_2014"
arcpy.MakeRasterLayer_management("Streets2014_Intrsct_Y_raster", lyr_all)
inRaster = lyr_all

# Execute FocalStatistics
```

```
outFocalStatistics = FocalStatistics(inRaster, neighborhood_400, "SUM","")
outFocalStatistics.save(out_workspace+"Rdsfty_2014_ALL_400m")
print inRaster
outFocalStatistics = FocalStatistics(inRaster, neighborhood_1600, "SUM","")
outFocalStatistics.save(out_workspace+"Rdsfty_2014_ALL_1600m")
print inRaster

# Low Mobility Streets
# Set focal statistics variables
lyr_lowmob = "lowmob_2014"
arcpy.MakeRasterLayer_management("Streets2014_LOW_MBLTY_raster", lyr_lowmob)
inRaster = lyr_lowmob

# Execute FocalStatistics

outFocalStatistics = FocalStatistics(inRaster, neighborhood_400, "SUM","")
outFocalStatistics.save(out_workspace+"Rdsfty_2014_LOWMOB_400m")
print inRaster
outFocalStatistics = FocalStatistics(inRaster, neighborhood_1600, "SUM","")
outFocalStatistics.save(out_workspace+"Rdsfty_2014_LOWMOB_1600m")
print inRaster

'''
# Parking available Streets NOT AVAILABLE 2014
# Set focal statistics variables
lyr_parkavail = "pkavail_2013"
arcpy.MakeRasterLayer_management("Street2013_PKAVAIL_raster", lyr_parkavail)
inRaster = lyr_parkavail

# Execute FocalStatistics

outFocalStatistics = FocalStatistics(inRaster, neighborhood_400, "SUM","")
outFocalStatistics.save(out_workspace+"Rdsfty_2013_PKAVAIL_400m")
print inRaster
outFocalStatistics = FocalStatistics(inRaster, neighborhood_1600, "SUM","")
outFocalStatistics.save(out_workspace+"Rdsfty_2013_PKAVAIL_1600m")
print inRaster
'''
# Roundabout Streets
# Set focal statistics variables
lyr_rdabout = "rdabout_2014"
arcpy.MakeRasterLayer_management("Streets2014_RDABT_raster", lyr_rdabout)
inRaster = lyr_rdabout

# Execute FocalStatistics

outFocalStatistics = FocalStatistics(inRaster, neighborhood_400, "SUM","")
```

```
outFocalStatistics.save(out_workspace+"Rdsfty_2014_RDABT_400m")
print inRaster
outFocalStatistics = FocalStatistics(inRaster, neighborhood_1600, "SUM","")
outFocalStatistics.save(out_workspace+"Rdsfty_2014_RDABT_1600m")
print inRaster

# Divider Streets
# Set focal statistics variables
lyr_divider = "divider_2014"
arcpy.MakeRasterLayer_management("Streets2014_DIVIDER_raster", lyr_divider)
inRaster = lyr_divider

# Execute FocalStatistics

outFocalStatistics = FocalStatistics(inRaster, neighborhood_400, "SUM","")
outFocalStatistics.save(out_workspace+"Rdsfty_2014_DIVIDER_400m")
print inRaster
outFocalStatistics = FocalStatistics(inRaster, neighborhood_1600, "SUM","")
outFocalStatistics.save(out_workspace+"Rdsfty_2014_DIVIDER_1600m")
print inRaster

end = time.time()
print str((end-start)/60)+" minutes"
```