Table of Contents

[Current Data Preparation for Raster Analysis: 3](#_Toc483475907)

[Source of Vector Files: 3](#_Toc483475908)

[Vector Files Clean-up: 4](#_Toc483475909)

[Joining Intersection Attribute to Road Segment Geometry 4](#_Toc483475910)

[Calculate Lane per Direction and Marked Center Line 4](#_Toc483475911)

[Shapefile and Attribute 5](#_Toc483475912)

[Technical guide on R script: 8](#_Toc483475913)

[Section A: User Input 8](#_Toc483475914)

[Section B: Setting the Extent for the Study 9](#_Toc483475915)

[Section C: Bike Segment Evaluation w/o Intersection Approach: 10](#_Toc483475916)

[Section C1: Rasterize using gdalUtils library 10](#_Toc483475917)

[Section C2: Physical Separated Bike Lane 10](#_Toc483475918)

[Section C3: Creating Masking Raster Layer 11](#_Toc483475919)

[Section C4: Bike Route with or w/o Adjacent Parking 11](#_Toc483475920)

[Section C5: Urban/Suburban Mixed Use Traffic Criteria 12](#_Toc483475921)

[Section C6: Sharrow Criteria 12](#_Toc483475922)

[Section C7: Combining Bike Lane Score and Mixed Used Traffic Score 13](#_Toc483475923)

[Section D: Right Turn Lane Criteria: 14](#_Toc483475924)

[Section D1: Rasterizing Right Turn Lane Configuration, Length and Approach 14](#_Toc483475925)

[Section D2: Right Turn Lane Set Up Layer and Masks 14](#_Toc483475926)

[Section D3: Right Turn Lane Score Assignment 15](#_Toc483475927)

[Section E: Left Turn Lane Criteria 16](#_Toc483475928)

[Section E1: Rasterizing Left Turn Lane Criteria 16](#_Toc483475929)

[Section E2: Left Turn Lane Set up Layers and Masks 16](#_Toc483475930)

[Section E3: Left Turn Lane Score Assignment 17](#_Toc483475931)

[Section E4: Combine LTL Score 18](#_Toc483475932)

[Section F: Un-Signalized Crossing 19](#_Toc483475933)

[Section F1: Rasterize Un-Signalized Crossing Criteria 19](#_Toc483475934)

[Section F2: Un-signalized Crossing Set Up Layers and Masks 19](#_Toc483475935)

[Section F3: Un-signalized Intersection Crossing with Median 20](#_Toc483475936)

[Section F4: Un-signalized Intersection Crossing without Median 21](#_Toc483475937)

[Section G: Visualizing the Scores 22](#_Toc483475938)

[Section G1: Plotting the Raster Score Layer 22](#_Toc483475939)

[Section H: Detect Island of Low Stress Area 23](#_Toc483475940)

[Section H1: Finding Cluster of Low Stress Area 23](#_Toc483475941)

[Section I: Route Calculation 24](#_Toc483475942)

[Section I1: Shortest Route with Lowest Stress Level 24](#_Toc483475943)

[Future Data Collection for BLTS and PLTS Analysis 25](#_Toc483475944)

# Current Data Preparation for Raster Analysis:

The BLTS analysis will be conducted in a raster format where all the attribute input and score output will be raster based format. This approach will allows CUUATS to integrate different models with open source format and has the same data type.

Current data are stored as vector (ESRI feature class/shapefile) format and therefore will need to be prepared before BLTS analysis can be performed. Currently, all the attributed required for the BLTS analysis are stored in 5 different geometries:

* UAB (CCGIS)
* StreetCL (CCGISC)
* Total Lane (CUUATS)
* Bike Lane and Pedestrian (CUUATS)
* Intersection Approach (CUUATS)

## Source of Vector Files:

**UAB**

Vector feature class stored in G:\Resources\Data\Boundary.gdb\UAB2013 which is the Urbanized Area Boundary in 2013, this will be used for the extent for our study

**StreetCL**

StreetCL is a vector feature class maintained by CCGISC under the path G:\Resources\Connections\CCGISV.sde\CCGISV.CCGIS.Transportation\CCGISV.CCGIS.StreetCL

**Total Lane**

Total Lane is a vector feature class maintained by CUUATS under the path G:\CUUATS\Local Accessibility and Mobility Analysis\Data\LocalAccessibilityAndMobilityAnalysis.gdb\totallanes

**Bike Lane and Pedestrian Path**

Bicycle and Pedestrian Path is a vector feature class maintained by CUUATS under the path G:\Resources\Connections\PCD\_DataCollection.sde\PCD.PCDQC.DataCollection\PCD.PCDQC.BicyclePedestrianath

**Intersection Approach**

Intersection Approach is a vector feature class maintained by CUUATS under the path G:\CUUATS\Local Accessibility and Mobility Analysis\Data\LocalAccessibilityandMobilityAnalysis.gbd\intersections\_blts\_Clip1

## Vector Files Clean-up:

Following the ODOT methodology, the BLTS analysis is a segment analysis where a high stress intersection approach will govern the score of the entire road segment. However, CUUATS is using a raster approach to the analysis, which loses the segment characteristic in this format. In order to address this problem, the attributes needed for the analysis are joined into the same geometry before the rasterization process.

### Joining Intersection Attribute to Road Segment Geometry

Software: ArcGIS

Input Data: StreetCL, Intersection

Purpose: Distributing intersection data to the appropriate road segment

1. Use Buffer function to buffer Intersections – 30 feet
2. Use Intersect function to intersect buffers with StreetCL, create a point output: buffer\_int\_points
3. Using nearest table function to calculate the degree of buffer\_int\_points with respect to original intersections. The result is a tables containing degrees.
4. Join buffer\_int\_points with nearest tables. The output is the buffer\_int\_points with degrees field.
5. Reclassify degrees into four categories. Use the reclass function found in Script.py in the same folder.
6. Use field calculator to recalculate each direction using Calculate function found in Script.py in the same folder.

### Calculate Lane per Direction and Marked Center Line

Software: ArcGIS

Input Data: Total Lane

Purpose: Calculate lane per direction from total lane

1. Use the field calculator function to divide total lane by two and round down the result (ex. 5/2 = 2, 3/2 = 1)

Purpose: Determine if the road segment has marked center line

1. Manually go through images and marked rather the road segment has marked center line.

### Shapefile and Attribute

For the purpose of using the gdal\_rasterize function in R script, geodatabase feature classes are exported as shapefiles for the analysis.

All the relevant attribute fields are all in short integer field and converted to short integer field to the rasterization process.

#### Bike Lane

"L:/Sustainable Neighborhoods Toolkit/TIFF/bikeLane.shp"

1. **Bike Lanes Classification(“PathType”):** The attribute containing physically separated bike lanes based on the BikePedestrianPath vector file. The value represent the classification of the bike lanes

|  |  |
| --- | --- |
| **Value** | **Definition** |
| 1 | Shared-Use Path (sidepath) |
| 2 | Divided Shared-Use Path |
| 3 | Shared-Use Path (off-street) |
| 4 | Bike Path |
| 5 | UIUC Bike Path |
| 6 | Bike Lanes (on-street) |
| 7 | Walking Path |
| 8 | Shared Lane Markings (sharrows) |
| 9 | Shared Bike/Parking Lanes |
| 10 | Bike Route |
| 11 | Trail Closed |

1. **Width of bike lanes(“Width”):** The attribute contains the width of the bike lane. The value is the width(ex. 7’, 5.5’). All the other area is represented by NA.
2. **Parking lane width(“pkLaneWidt”):** The attribute contains the width of the parking lane. The value is the width.
3. **Combing parking and bike lane width(“bikeParkWidth”):** The attribute contains the combine width of parking and bike lane.
4. **Bike lane with adjacent parking lane criteria(“hasParki\_1”):** The attribute contains a value of 1 or 0, where 1 represent that road segment has adjacent parking while 0 represents it does not has adjacent parking

#### Total Lane

"L:/Sustainable Neighborhoods Toolkit/TIFF/TotalLane.shp"

1. **Lane per direction (“lanePerDir”):**  The attribute contains the number of lane per direction for the road segment.

|  |  |
| --- | --- |
| **Value** | **Definition** |
| 0 | Unmarked Center Lane |
| 1 | 1 lane per direction |
| 2 | 2 lane per direction |
| 3 | 3 lane per dicection |

#### StreetCL

"L:/Sustainable Neighborhoods Toolkit/TIFF/StreeCL.shp"

1. **Speed (“SPEED”):** The attribute contains the speed limit for the road segment.
2. **Right turn lane configuration North (“RLT\_Conf\_N”):** The attribute contains the type of right turn lane configuration at the north end of the road segment.
3. **Right turn lane configuration South (“RLT\_Conf\_S”):** The attribute contains the type of right turn lane configuration at the south end of the road segment.
4. **Right turn lane configuration East (“RLT\_Conf\_E”):** The attribute contains the type of right turn lane configuration at the east end of the road segment.
5. **Right turn lane configuration West (“RLT\_Conf\_W”):** The attribute contains the type of right turn lane configuration at the west end of the road segment.

|  |  |
| --- | --- |
| **Value** | **Definition** |
| 0 | No right turn lane |
| 1 | Single |
| 2 | Dual Exclusive / Share |

1. **Bike lane approach (“BL\_Appr\_Al”):** the attribute contains the type of bike lane approach alignment

|  |  |
| --- | --- |
| **Value** | **Definition** |
| 0 | NA |
| 1 | Straight |
| 2 | Ends |
| 3 | Drop |

1. **Right turn lane length (“RTL\_Length”):** the attribute contains the length of right turn length(ft)
2. **Left turn lane lane-crossed North (“LTL\_lanesc”):** the attribute contains the number of lane a bicyclist has to cross to reach the left turn lane at the north end of the road segment.
3. **Left turn lane lane-crossed South (“LTL\_lane\_1”):** the attribute contains the number of lane a bicyclist has to cross to reach the left turn lane at the south end of the road segment.
4. **Left turn lane lane-crossed East (“LTL\_lane\_2”):** the attribute contains the number of lane a bicyclist has to cross to reach the left turn lane at the east end of the road segment.
5. **Left turn lane lane-crossed West (“LTL\_lane\_3”):** the attribute contains the number of lane a bicyclist has to cross to reach the left turn lane at the west end of the road segment.
6. **Left turn lane configuration North (“LTL\_Conf\_N”):** the attribute contains the type of left turn lane configuration at the north end of the road segment.
7. **Left turn lane configuration South (“LTL\_Conf\_S”):** the attribute contains the type of left turn lane configuration at the south end of the road segment.
8. **Left turn lane configuration East (“LTL\_Conf\_E”):** the attribute contains the type of left turn lane configuration at the east end of the road segment.
9. **Left turn lane configuration West (“LTL\_Conf\_W”):** the attribute contains the type of left turn lane configuration at the west end of the road segment.

#### Intersection

1. **Turn lane crossed at the North South direction (“TotalLanes”):** the attribute contains the number of lanes that need to be crossed if crossing from the North-South direction.
2. **Turn lane crossed at the East West direction (“TotalLan\_1”):** the attribute contains the number of lanes that need to be crossed if crossing from the East-West direction.
3. **Signalized intersection (“signal”):** the attribute contains rather the intersection has a signalized control. 1 – Yes, 0 – No
4. **Median present (“med\_ref”):** the attribute contains rather the intersection has a median present. 1 – Yes, 0 – No

# BLTS Methodology:

The analysis are broken down into 7 steps, physically separated bike lanes, bike lane, urban/suburban mixed traffic, right turn lane, left turn lane, un-signalized intersection crossing. It is a step by step process to develop a score for each criteria. Each new score will be combined with the previous score generate a new score.

# Technical guide on R script:

For our analysis, the input geometries and attributed are from ESRI’s shapefile format. All the data are imported into R for all the future steps in the process.

This document describe the process of the SustainableNeighbor.R script.

## Section A: User Input

Variables in this section can be adjusted to change the path to certain files and adjust for the setting in the analysis.

1. Set the path to geodatabase that contains feature classes that are important for analysis (Intersection point data)
2. Setting the working directory where output files are stored
3. Set the resolution for raster cells, the default setting is 100. Value of 100 represents the cell size of 100x100 in all of the raster analysis
4. Set the path to geodatabase that contains the feature class that are important for analysis (Urbanized Area Boundary) – to set the extent of our data
5. Set the directory path the stores the shapefiles. These are used in gdal\_rasterize process later

6,7,8,9 Set the name of the shapefiles

10. Set the projection system for the study

11. Optional: Set a no path value for the study

## Section B: Setting the Extent for the Study

Use the readOGR function feature class from geodatabase, and extract the extent from the feature class for analysis.

|  |  |
| --- | --- |
| 1  2  3  4  5 | #Read Boundary for the study area  UA <- readOGR(dsn=boundary.fgdb, layer="UAB2013")  #Set Extent for Test Area  extent<-extent(UA) |

## Section C: Bike Segment Evaluation w/o Intersection Approach:

### Section C1: Rasterize using gdalUtils library

* 2. Set the source path to the shapefile
* 5. Create an empty raster, set the extent, resolution, and projection
* 6. Write an GeoTiff file in the directory for the rasterization process
* 7. Use the gdal\_utility library and rasterize the shapefile into raster format
* 9. Set the projection system for the new raster layer

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | #Set path to Bike Lane shapefile  src\_datasource <- paste(shape.path,bike.name, sep = "")  #Rasterize Path Type  r.bikelane <- raster(ext=extent, resolution = resolution, crs = crs)  bikelane.tif <- writeRaster(r.bikelane, filename = "bikelane.tif", format="GTiff", overwrite=TRUE)  facility.raster <- gdal\_rasterize(src\_datasource = src\_datasource, dst\_filename = "bikelane.tif", a="PathType", at=TRUE, output\_Raster = TRUE)  crs(facility.raster) <- crs |

### Section C2: Physical Separated Bike Lane

**Classification method are made for the physically separated paths and lanes:**

* 1. Create an empty raster to store the score with approximate extent, resolution and coordinate system
* 2. Assign a score for empty cells
* 4-8. Create mask layers that which contains the following path type offstreet (Path type: 1, 2, 3, 4, 5, 10)
* 10. Since these biking facilities are usually low stress and off-road, and will be assigned LTS 1.

|  |  |
| --- | --- |
| 1  2  3  4  5  5  6  7  8  9  10 | scoreBike <- raster(ext=extent, resolution = resolution, crs = crs)  scoreBike[] <- npv  osft1 <- facility.raster == 1  osft2 <- facility.raster == 2  osft3 <- facility.raster == 3  osft4 <- facility.raster == 4  osft5 <- facility.raster == 5  osft10 <- facility.raster == 10  scoreOffStreet[osft1 | osft2 | osft3 | osft4 | osft5 | osft10] <- 1 |

### Section C3: Creating Masking Raster Layer

Creating masking raster layer that will be used for the analysis

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | #Bike Criteria (Does the bike lane has adj parking? 1 - yes, 0 - no)  ly <- bikeCrit.raster == 1  ln <- bikeCrit.raster == 0  #Lane per direction (for Biking facilities)  lpd1 <- lanePerDir.raster == 1 | lanePerDir.raster == 0  lpd2 <- lanePerDir.raster >= 2 |

### Section C4: Bike Route with or w/o Adjacent Parking

Use all the mask layers, and Exhibit 14-3 to assign score to bike route with adjacent parking

Line 3 create a mask for all on street bike lane. Lane 6-9 combine the criteria for on street bike lane, **has adjacent parking**, lane per direction, speed limit, and bike land plus parking lane width.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | #Bike Lane with Adjacent Parking Lane Criteria  #Bike Path Type  bikelane <- facility.raster == 6 | facility.raster == 9  #Set the default score for all biking facilities  scoreBike[bikelane & ly & lpd1 & sp25 & bpw15] <- 1  scoreBike[bikelane & ly & lpd1 & sp30 & bpw15] <- 1  scoreBike[bikelane & ly & lpd1 & sp35 & bpw15] <- 2  scoreBike[bikelane & ly & lpd1 & sp40 & bpw15] <- 2 |

Use all the mask layers and Exhibit 14-4 to assign score to bike route without Adjacent Parking

Line 1-3 combine the criteria for on street bike lane, **does not have adjacent parking**, lane per direction, speed limit, and bike land plus parking lane width.

|  |  |
| --- | --- |
| 1  2  3 | scoreBike[bikelane & ln & lpd1 & spless30 & bwgreat7] <- 1  scoreBike[bikelane & ln & lpd1 & sp35 & bwgreat7] <- 2  scoreBike[bikelane & ln & lpd1 & sp40 & bwgreat7] <- 3 |

### Section C5: Urban/Suburban Mixed Use Traffic Criteria

Line 1 and 2 create a new layer to store the mixed use traffic score

Line 4 to 5 assign score based on Exhibit 14-5 using speed and lane per direction criteria

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | scoreMix <- raster(ext=extent, resolution = resolution, crs = crs)  scoreMix[] <- npv  scoreMix[sp25 & lane0] <- 1  scoreMix[sp30 & lane0] <- 2  scoreMix[spgreat35 & lane0] <- 3 |

### Section C6: Sharrow Criteria

Line 2 creates a mask for segment that has sharrow present

Line 3-5 creates a raster layers that assign sharrow with a -1 value, and everything else contain a zero value.

Line 7 add the mixed use score layer to the sharrow layer

Line 8 if the BLTS in the original mixed used layer is already 1, sharrow will not reduce the score to zero

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | #No bike Lane but has sharrow  sharrow <- facility.raster == 8  sharrow[sharrow != 1] <- 0  sharrow[is.na(sharrow)] <- 0  sharrow[sharrow == 1] <- -1  scoreMix <- scoreMix + sharrow  scoreMix[scoreMix == 0] <- 1 |

### Section C7: Combining Bike Lane Score and Mixed Used Traffic Score

Line 2-3 creates a new layer to store the combine score of bike lane and mixed used traffic

Line 4 creates a raster stack of the two raster layer

Line 5 overlays the two raster layers and select the minimum score from the two layers

Line 8-9 writes the raster onto the local drive with a GeoTIFF format

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | ##Calculate Combine Score for bike facilities and mix used traffic  scoreComb <- raster(ext=extent, resolution = resolution, crs = crs)  scoreComb[] <- npv  stk <- stack(scoreMix,scoreBike)  scoreComb <- overlay(stk, fun=min)  ###Export score as a GeoTiff  filename <- paste("scoreNoInt", resolution)  writeRaster(scoreComb, filename, format = "GTiff", overwrite=TRUE) |

## Section D: Right Turn Lane Criteria:

### Section D1: Rasterizing Right Turn Lane Configuration, Length and Approach

Line 2 sets the path to the shapefile

Line 3 creates a new raster with appropriate extent, resolution, and projection system\

Line 4 creates a new GeoTIFF file for the gdal\_rasterize process

Line 7 rasterizes the shapefile into the raster file

Line 9 set the projected coordinate system for the new raster layer

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9 | #Rasterize Right Turn Lane Configuration (Section D1)  src\_datasource <- paste(shape.path,StreetCL.name, sep = "")  r.RTL\_Conf\_N <- raster(ext=extent, resolution = resolution, crs = crs)  RTL\_Conf\_N.tif <- writeRaster(r.RTL\_Conf\_N, filename = "RTL\_Conf\_N", format="GTiff", overwrite=TRUE)  RTL\_Conf\_N.raster <- gdal\_rasterize(src\_datasource, dst\_filename = "RTL\_Conf\_N.tif", a="RTL\_Conf\_N",at=TRUE,output\_Raster = TRUE)  crs(RTL\_Conf\_N.raster) <- crs |

### Section D2: Right Turn Lane Set Up Layer and Masks

Creates a scoreRLT raster layer and a temporary layer to store the score, due to the fact there might be multiple directions at the intersection

|  |  |
| --- | --- |
| 1  2  3  4  5 | scoreRTL <- raster(ext=extent, resolution = resolution, crs = crs)  scoreRTL[] <- 0  scoreRTL.temp <- raster(ext=extent, resolution = resolution, crs = crs)  scoreRTL.temp[] <- 0 |

Create the masks layer for the Right Turn Lane Criteria

**Bike Lane Approach Alignment:** Straight = 1, Ends = 2, Drop = 3, N/A= 0

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10 | #create mask layer for RTL length  RTL\_Length\_less150 <- RTL\_Length.raster <= 150 & RTL\_Length.raster > 0  RTL\_Length\_great150 <- RTL\_Length.raster > 150  RTL\_Length\_any <- RTL\_Length.raster > 0  #create layer for the Right-turn lane approach configuration  RTL\_Appro\_Str <- BL\_Appr\_Al.raster == 1  RTL\_Appro\_Lef <- BL\_Appr\_Al.raster == 2 | BL\_Appr\_Al.raster == 3  RTL\_Appro\_Any <- BL\_Appr\_Al.raster == 1 | BL\_Appr\_Al.raster == 2 | BL\_Appr\_Al.raster == 3 | BL\_Appr\_Al.raster == 0 |

### Section D3: Right Turn Lane Score Assignment

This follows Exhibit 14-7 Right Turn Lane Criteria to assign more to each segment of road with right turn lane criteria

Line 3 create a rasterStack object (collection of raster layers) from the four right turn lane configuration for the different direction at the intersection

Line 4 assigns name to the four layers

Line 6 loops through the four layers and create a RTL score

Line 7-8 creates a masking layer for right-turn lane configuration for each direction

Line 10-13 assigns the score based on right turn lane configuration, right-turn lane length, bike lane approach alignment to the temporary layer

Line 14-15 combines the two temporary score with the overall score and the new score will be the layer with the highest number (higher stress level)

Line 20-21 changes all the previous no path value to zero

Line 22-23 creates a rasterStack and overlay the two layers and select the max value from the two

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23 | #Loop through the RTL Configuration for N, S, E, W and assign score for each direction of the intersection  #Combine the layers into rasterStack  int\_Dir <- stack(RTL\_Conf\_N.raster, RTL\_Conf\_S.raster,RTL\_Conf\_E.raster,RTL\_Conf\_W.raster)  names(int\_Dir) <- c("North", "South", "East", "West")  for (i in 1:nlayers(int\_Dir)) {  RTL\_Conf\_S <- int\_Dir[[i]] == 1  RTL\_Conf\_DE\_S <- int\_Dir[[i]] == 2 | int\_Dir[[i]] == 1  title = names(int\_Dir[[i]])  scoreRTL.temp[RTL\_Conf\_DE\_S & RTL\_Length\_any & RTL\_Appro\_Any] <- 4  scoreRTL.temp[RTL\_Conf\_S & RTL\_Length\_great150 & RTL\_Appro\_Str] <- 3  scoreRTL.temp[RTL\_Conf\_S & RTL\_Length\_any & RTL\_Appro\_Lef] <- 3  scoreRTL.temp[RTL\_Conf\_S & RTL\_Length\_less150 & RTL\_Appro\_Str] <- 2  scoreRTL <- stack(scoreRTL, scoreRTL.temp)  scoreRTL <- overlay(scoreRTL, fun=max)  }  #Overlay the Score w/o intersection and with RTL criteria, assign null value to zero and select the maximum from the two  scoreComb[scoreComb == 5] <- 0  scoreRTL[scoreRTL== 5] <- 0  Comb\_RTL <- stack(scoreComb, scoreRTL)  scoreCombRTL <- overlay(Comb\_RTL, fun = max) |

## Section E: Left Turn Lane Criteria

### Section E1: Rasterizing Left Turn Lane Criteria

Rasterizing the all the criteria for left turn lane

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | #Lane Cross  src\_datasource <- paste(shape.path,StreetCL.name, sep = "")  r.LTL\_lanesc <- raster(ext=extent, resolution = resolution, crs = crs)  LTL\_lanesc.tif <- writeRaster(r.LTL\_lanesc, filename = "LTL\_lanesc", format="GTiff", overwrite=TRUE)  LTL\_lanesc.raster <- gdal\_rasterize(src\_datasource, dst\_filename = "LTL\_lanesc.tif", a="LTL\_lanesc",at=TRUE,output\_Raster = TRUE)  crs(LTL\_lanesc.raster) <- crs |

### Section E2: Left Turn Lane Set up Layers and Masks

Setting up score and a temporary raster layers

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | #Create empty raster to store the score for LTL (Section E2)  scoreLTL <- raster(ext=extent, resolution = resolution, crs = crs)  scoreLTL[] <- 0  scoreLTL.temp <- raster(ext=extent, resolution = resolution, crs = crs)  scoreLTL.temp[] <- 0 |

### Section E3: Left Turn Lane Score Assignment

Line 2 creates a rasterStack for LTL configuration for each direction

Line 3 creates a rasterStack for LTL lane crossed for each direction

Line 6 loops through the four direction in the raster stack

Line 8-11 creates a mask for number of lanecrossed or exclusive left turn lanes

Line 14-27 assigned score based on Exhibit 14-8 criteria

Line 29-30 creates a raster stack of the score and temporary layer, and overlay them based on maximum value

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31 | #Create Stack for LTL Criteria  LTL\_Conf\_Dir <- stack(LTL\_Conf\_N.raster, LTL\_Conf\_S.raster,LTL\_Conf\_E.raster,LTL\_Conf\_W.raster)  LTL\_LC\_Dir <- stack(LTL\_lanesc.raster, LTL\_lane\_1.raster, LTL\_lane\_2.raster, LTL\_lane\_3.raster)  #Create evaluation layer for lane crossed in one intersection  for(i in 1:nlayers(LTL\_LC\_Dir)) {  #Setting criteria for evaluation  laneCrossed\_0 <- LTL\_LC\_Dir[[i]] == 0  laneCrossed\_1 <- LTL\_LC\_Dir[[i]] == 1  laneCrossed\_2 <- LTL\_LC\_Dir[[i]] >= 2  laneCrossed\_DE <- LTL\_Conf\_Dir[[i]] != 0  #Scoring based on Exhibit 14-8 Left Turn Lane Criteria  scoreLTL.temp[sp25 & laneCrossed\_0] <- 2  scoreLTL.temp[sp25 & laneCrossed\_1] <- 2  scoreLTL.temp[sp25 & laneCrossed\_2] <- 3  scoreLTL.temp[sp25 & laneCrossed\_DE] <- 4    scoreLTL.temp[sp30 & laneCrossed\_0] <- 2  scoreLTL.temp[sp30 & laneCrossed\_1] <- 3  scoreLTL.temp[sp30 & laneCrossed\_2] <- 4  scoreLTL.temp[sp30 & laneCrossed\_DE] <- 4    scoreLTL.temp[spgreat35 & laneCrossed\_0] <- 3  scoreLTL.temp[spgreat35 & laneCrossed\_1] <- 4  scoreLTL.temp[spgreat35 & laneCrossed\_2] <- 4  scoreLTL.temp[spgreat35 & laneCrossed\_DE] <- 4    scoreLTL <- stack(scoreLTL, scoreLTL.temp)  scoreLTL <- overlay(scoreLTL, fun=max)  } |

### Section E4: Combine LTL Score

Create a new score layer, overlay the previous score with the LTL score based on maximum value

|  |  |
| --- | --- |
| 1  2  3  4  5 | #Combinging the Score with Mixed Used, Bike Lane, RLT and LTL (Section E4)  score.Comb.RLT.LTL <- raster(ext=extent, resolution = resolution, crs = crs)  score.Comb.RLT.LTL[] <- 0  score.Comb.RLT.LTL <- stack(scoreComb, scoreLTL)  score.Comb.RLT.LTL <- overlay(score.Comb.RLT.LTL, fun = max) |

## Section F: Un-Signalized Crossing

### Section F1: Rasterize Un-Signalized Crossing Criteria

Line 3-6 reads the features class from geodatabase, creates an empty raster, use the R rasterize function to rasterize the speed attributed, when two speed exist at the intersection, the higher speed will be used.

Line 9-15 gdal\_rasterize process to rasterize all the criteria needed for un-signalized crossing

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15 | #Median Criteria (Section F)  #rasterize maxspeed  Street <- readOGR(dsn=path.fgdb, layer = "Street\_w\_Int\_Clip")  maxsp <- raster(ext=extent, resolution = resolution, crs = crs)  maxsp <- rasterize(Street, maxsp, field="SPEED", fun='max')  crs(maxsp) <- crs  #rasterize intersections total lanecrossed  src\_datasource <- paste(shape.path,Int.name, sep = "")  r.totallanes\_ns <- raster(ext=extent, resolution = resolution, crs = crs)  totallanes\_ns.tif <- writeRaster(r.totallanes\_ns, filename = "totallanes\_ns", format="GTiff", overwrite=TRUE)  totallanes\_ns.raster <- gdal\_rasterize(src\_datasource, dst\_filename = "totallanes\_ns.tif", a="TotalLanes",at=TRUE,output\_Raster = TRUE)  crs(totallanes\_ns.raster) <- crs |

### Section F2: Un-signalized Crossing Set Up Layers and Masks

Set up empty score layer, temporary layer, and masks layer

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | #create an empty score layer for Median criteria  scoreMed <- raster(ext=extent, resolution = resolution, crs = crs)  scoreMed[] <- 0  scoreMed.temp <- raster(ext=extent, resolution = resolution, crs = crs)  scoreMed.temp[] <- 0  #Criterias for Median  median.true <- median.raster == 1  median.false <- median.raster == 0  #mask for speed  sp25 <- maxsp <= 25  sp30 <- maxsp == 30  sp35 <- maxsp == 35  sp40 <- maxsp >= 40  #mask for unsignalized intersection  unsignal <- signal.raster == 0 |

### Section F3: Un-signalized Intersection Crossing with Median

Line 2 creates a rasterStack for total lanes crossed for NS and EW

Line 5 loops through the two directional total lanes layer

Line 6-8 creates masks for total lane crossed masks

Line 10-24 assigns score based on Exhibit 14-9

Line 26-27 overlays temp score with score layer using the max function

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28 | #stack the total lane ns and ew together  tl\_stack <- stack(totallanes\_ns.raster, totallanes\_ew.raster)  #for loop for intersection that has median  for (i in 1:nlayers(tl\_stack)) {  tlc3 <- tl\_stack[[i]] <= 3  tlc45 <- tl\_stack[[i]] == 4 | tl\_stack[[i]] == 5  tlc6 <- tl\_stack[[i]] >= 6    scoreMed.temp[unsignal & median.true & sp25 & tlc3] <- 1  scoreMed.temp[unsignal & median.true & sp25 & tlc45] <- 2  scoreMed.temp[unsignal & median.true & sp25 & tlc6] <- 4    scoreMed.temp[unsignal & median.true & sp30 & tlc3] <- 1  scoreMed.temp[unsignal & median.true & sp30 & tlc45] <- 2  scoreMed.temp[unsignal & median.true & sp30 & tlc6] <- 4    scoreMed.temp[unsignal & median.true & sp35 & tlc3] <- 2  scoreMed.temp[unsignal & median.true & sp35 & tlc45] <- 3  scoreMed.temp[unsignal & median.true & sp35 & tlc6] <- 4    scoreMed.temp[unsignal & median.true & sp40 & tlc3] <- 3  scoreMed.temp[unsignal & median.true & sp40 & tlc45] <- 4  scoreMed.temp[unsignal & median.true & sp40 & tlc6] <- 4    scoreMed <- stack(scoreMed, scoreMed.temp)  scoreMed <- overlay(scoreMed, fun=max)  } |

### Section F4: Un-signalized Intersection Crossing without Median

Line 2 loops through the total lane crossed layers

Line 3-4 creates masks for the criteria

Line 6-20 assigns score based on Exhibit 140-10

Line 22-23 combines score based on maximum value

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24 | #for loop for intersection that does not has median (Section F4)  for (i in nlayers(tl\_stack)) {  tlc12 <- tl\_stack[[i]] == 1 | tl\_stack[[i]] == 2  tlcGreat4 <- tl\_stack[[i]] >= 4    scoreMed.temp[unsignal & median.false & sp25 & tlc12] <- 1  scoreMed.temp[unsignal & median.false & sp25 & tlc3] <- 1  scoreMed.temp[unsignal & median.false & sp25 & tlcGreat4] <- 2    scoreMed.temp[unsignal & median.false & sp30 & tlc12] <- 1  scoreMed.temp[unsignal & median.false & sp30 & tlc3] <- 2  scoreMed.temp[unsignal & median.false & sp30 & tlcGreat4] <- 3    scoreMed.temp[unsignal & median.false & sp35 & tlc12] <- 2  scoreMed.temp[unsignal & median.false & sp35 & tlc3] <- 3  scoreMed.temp[unsignal & median.false & sp35 & tlcGreat4] <- 4    scoreMed.temp[unsignal & median.false & sp40 & tlc12] <- 3  scoreMed.temp[unsignal & median.false & sp40 & tlc3] <- 4  scoreMed.temp[unsignal & median.false & sp40 & tlcGreat4] <- 4    scoreMed <- stack(scoreMed, scoreMed.temp)  scoreMed <- overlay(scoreMed, fun=max)  } |

## Section G: Visualizing the Scores

### Section G1: Plotting the Raster Score Layer

Create break points for the symbolization, assign values not relevant to the map to NA, set the title, and create the plot

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | #Plot only biking facilities  breakpoints <- c(0,1,2,3,4)  colors <- c("blue","green","orange","red")  scoreBike[scoreBike == 5] <- NA  title <- paste("Score with only biking faciliities - Res:", resolution)  plot(scoreBike,breaks=breakpoints,col=colors, main =title)  #Plot Mix traffic w/o Int  breakpoints <- c(0,1,2,3,4)  colors <- c("blue","green","orange","red")  scoreComb[scoreComb == 0] <- NA  title <- paste("Combine Score w/o Intersection - Res:", resolution)  plot(scoreComb,breaks=breakpoints,col=colors, main =title) |

## Section H: Detect Island of Low Stress Area

### Section H1: Finding Cluster of Low Stress Area

Line 2 creates a new raster object from the combine score

Line 3 assigns cell that has a 3 or 4 to NA

Line 5 find cluster of cell that contain the value of 1 or 2

|  |  |
| --- | --- |
| 1  2  3  4  5  6 | #Detect Island of activities  score12 <- score.Comb.RLT.LTL  score12[score12 == 3 | score12 == 4] <- NA  plot(score12)  c.score12 <- clump(score12, directions = 8)  plot(c.score12, main = "Island of score of 1 and 2") |

## Section I: Route Calculation

### Section I1: Shortest Route with Lowest Stress Level

Line 1 copies a new copy of the overall score

Line 2 changes all the NA value to 9999

Line 3 creates a transition layer that is used later for shortest path calculation

Line 5 makes geoCorrection to the transition layer

Line 8-9 creates two points

Line 10 calculates the shortest path between the two points using the corrected transition layer and out put a formal spatial lines class

Line 14-17 converts the spatial line into spatial line data frame and read it as an ESRI shapefile

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17 | scoretest <- scoreALL  scoretest[is.na(scoretest)] <- 9999  tr1 <- transition(scoretest, function(x) 1/mean(x), direction=4)  plot(scoretest)  tr1C <- geoCorrection(tr1)  #commuteDistance(tr1C, testpoint)  C <- c(1000000,1240000)  U <- c(1030000,1260000)  CtoU <- shortestPath(tr1C, C, U, output="SpatialLines")  crs(CtoU) <- crs  plot(CtoU, add=TRUE)  df <- data.frame(id = c(1,2))  CtoU <- SpatialLinesDataFrame(CtoU, df)  dir.create(tempdir)  writeOGR(CtoU, dsn="tempdir", layer="CtoU", driver="ESRI Shapefile", overwrite\_layer = TRUE) |

# Future Data Collection for BLTS and PLTS Analysis