Kentucky Transportation Center

Research Report
KTC-15-11/SPR14-480-1F
DOI: http://dx.doi.org/10.13023/KTC.RR.2015.11

Spatial Database for Intersections

Our Mission

We provide services to the transportation community through research, technology transfer and education. We create and participate in partnerships to promote safe and effective transportation systems.

Kentucky Transportation Center
176 Oliver H. Raymond Building
Lexington, KY 40506-0281
(859) 257-4513
fax (859) 257-1815

www.ktc.uky.edu

Research Report
KTC-15-11/SPR14-480-1F

Spatial Database For Intersections

By

Eric R. Green
Research Engineer

Chris Blackden
Research Analyst

and

Michael A. Fields
Research Analyst

Kentucky Transportation Center
College of Engineering
University of Kentucky
Lexington, Kentucky

in cooperation with
Transportation Cabinet
Commonwealth of Kentucky

and

Federal Highway Administration
U.S. Department of Transportation

August 2015

1. Report No. KTC-15-11/SPR14-480-1F	2. Government Accession No.	3. Recipient's Catalog No
4. Title and Subtitle **Spatial Database For Intersections**		5. Report Date August 2015 6. Performing Organization Code
7. Author(s): Eric R. Green, Chris Blackden, Michael A. Fields		8. Performing Organization Report No. KTC-15-11/SPR14-480-1F
9. Performing Organization Name and Address Kentucky Transportation Center College of Engineering University of Kentucky Lexington, Kentucky 40506-0281		10. Work Unit No. (TRAIS) 11. Contract or Grant No. KYSPR 14-480
12. Sponsoring Agency Name and Address Kentucky Transportation Cabinet 200 Mero Street Frankfort, Kentucky 40622		13. Type of Report and Period Covered 14. Sponsoring Agency Code

15. Supplementary Notes
Prepared in cooperation with the Kentucky Transportation Cabinet and the Federal Highway Administration

16. Abstract

Deciding which intersections in the state of Kentucky warrant safety improvements requires a comprehensive inventory with information on every intersection in the public roadway network. The Kentucky Transportation Cabinet (KYTC) had previously catalogued only those intersections where state-maintained roadways met. However, this inventory did not account for intersections between state- and locally-maintained routes, nor was it designed to accommodate regular updates. As such, the Kentucky Transportation Center (KTC) at the University of Kentucky developed a methodology to create and maintain a full inventory of every intersection in the state. The database contains precise location information as well as several safety and operational attributes for each point of an intersection. By replicating the topology factors used in the *Highway Safety* Manual (HSM), the research team categorized every intersection type, and developed. Safety Performance Functions (SPF) for each intersection type. The SPFs were used to rank each intersection. It is anticipated that this project's deliverables will be used to increase KYTC's ability to effectively allocate funds to maintain and improve intersection safety. Making the database available to expert users will allow continuous improvements. In the future, AADT data and traffic control information could be included.

17. Key Words	18. Distribution Statement		
Intersection, spatial database, safety performance function, nodes, cross intersection, tee intersection, approaches, legs	Unlimited, with approval of the Kentucky Transportation Cabinet		
19. Security Classification (report) Unclassified	**20. Security Classification (this page)** Unclassified	**21. No. of Pages** 23	**22. Price**

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

EXECUTIVE SUMMARY

Deciding which intersections in the state of Kentucky warrant safety improvements requires a comprehensive inventory with information on every intersection in the public roadway network. The Kentucky Transportation Cabinet (KYTC) had previously catalogued only those intersections where state-maintained roadways met. However, this inventory didn't account for intersections between state- and locally-maintained routes, nor was it designed to accommodate regular updates. As such, the Kentucky Transportation Center (KTC) at the University of Kentucky developed a methodology to create and maintain a full inventory of every intersection in the state. The database contains precise location information as well as several safety and operational attributes for each point of an intersection. By replicating the topology factors used in the *Highway Safety* Manual (HSM), the research team categorized every intersection type, and developed Safety Performance Functions (SPF) for each intersection type. The SPFs were used to rank each intersection. It is anticipated that this project's deliverables will be used to increase KYTC's ability to effectively allocate funds to maintain and improve intersection safety. Making the database available to expert users will allow continuous improvements. In the future, AADT data and traffic control information could be included.

1. INTRODUCTION

1.1 Background and Objectives

Intersections are one of 10 areas of emphasis listed in Kentucky's most recent Strategic Highway Safety Plan (SHSP). Crashes that occur at intersections represent approximately 25% of all collisions in Kentucky (Kentucky Traffic Collision Facts). Representatives from the Kentucky Transportation Cabinet (KYTC) have expressed that a maintainable intersection database is needed to reduce the number of crashes. Because state transportation agencies have become increasingly reliant on spatially explicit data, a spatially-enabled intersection database would benefit many areas of transportation. In 2003, the Kentucky Transportation Center (KTC) conducted a research study on vehicle crashes at intersections. The dataset produced during that research has since become the primary database used by the Cabinet for intersection safety analysis and prioritization ranking. Unfortunately, this database was created to analyze safety over a fixed time period and is now out-of-date.

The objective of this research study is to develop a comprehensive database that inventories all intersections in the state of Kentucky-- one that includes the points where all state-maintained and locally maintained roadways intersect. The major impetus for creating this database is to equip KYTC with the knowledge it needs to prioritize intersections for safety improvements. To that end, it is important to categorize intersections so they can be compared with one another. Comparisons must focus on similarly functioning sites instead of generating one priority list that ignores important geometric and operational differences. The database that resulted from this study can be updated using an automated process. It can identify any changes to roadway geometry that would alter the attributes of intersections. In addition to point features used for locating intersections, this database accounts for each intersection's zone of influence. This encompasses the area around intersection points at which crashes are assumed to be attributable to the intersection.

1.2 Literature Review

1.2.1 Classifying Intersections

Various methods have been developed to classify intersections and evaluate their safety. Simandl et al. (2015) used Google Maps/ Street View to collect visual observations of an intersection with a preset form or web portal. Observations were linked to an intersection node in spatial database, which were plotted using linear referencing. Simandl et al. limited their study to a small subset of intersections (non-signalized, state-route) in the state of Alabama. Campbell and Knapp (2005) discussed a more comprehensive intersection classification system in the state of Wisconsin. They used parameters such a traffic volume, number of legs, and presence of a median. These categories were then cross-referenced with crash data. Garber et al. (2011) expanded on earlier work by creating Safety Performance Functions (SPF) on the intersection classification system they developed for Virginia (although with far fewer categories than Campbell and Knapp used).

1.2.2 Safety Analysis

In recent analyses, intersections have been ranked using a procedure called critical rate analysis (Green, 2004). Critical rate analysis, however, is problematic for several reasons. First, it assumes a linear relationship between the number of crashes and traffic volume. Recent studies have determined that this is not a valid assumption. For example, sites with very low annual average daily traffic (AADT) that have experienced even one crash tend to have disproportionately high critical rates. For most roadway and intersection types, the relationship between the number of crashes and traffic volume is exponential (Srinivasan, 2011). The Highway Safety Manual (HSM) outlines procedures to develop SPFs that more

realistically represent the relationship between crashes and traffic volume. Unlike critical rate analysis, the SPFs also account for regression to the mean by using an Empirical Bayes statistical method.

1.2.3 Development of Safety Performance Functions

The HSM uses safety performance to recommend processes and guidelines for decision making. To properly apply the HSM procedures, SPFs should be developed after analysis of state-specific crash data. Developing SPFs by using historical data from local agencies should increase the accuracy and reliability of crash estimates. These estimates can help evaluate the potential effectiveness of alternative countermeasures. The HSM's regression models estimate the predicted average crash frequency for a site based on data from similar sites. After SPFs were developed for each site, they were adjusted using the Empirical Bayes (EB) approach to improve the accuracy of estimates and to address possible changes from regression to the mean.

2. METHODOLOGY

The spatial intersection database was created using ArcMap and Excel. The final product included a geodatabase with a polygon feature class called "Intersections" and a corresponding table containing classificatory attributes, including crash statistics. Selected intersections from this table – those satisfying certain data requirements explained below – were then used as inputs for the Safety Performance Function described in Chapter 3.

2.1 Preprocessing

2.1.1 Plotting the Node Usage Table

We began with a table provided by KYTC called "Kentucky Node Usage." The table is regularly updated, but for this pilot study, we used a version from November 2014. This table described nodes in the Kentucky road network. Nodes represent points where routes: 1) begin, 2) end, 3) intersect one another, or 4) change designation. Aside from intersections, route designations change most frequently at county boundaries. The same node, except in the case of a dead end route, was represented by multiple records in the table and was thus read by ArcGIS as multiple points "stacked" at the same spatial coordinate. For clarity, we referred to nodes as individual records and multi-record nodes as "node stacks." Stacked nodes, with some exceptions (see below), shared a unique identification number designated as "Knu_Node_ID" (knu = Kentucky node usage) in the node usage table. The number of nodes in a stack coincided with the number of approaches at that point. A dead-end has one node, a change in route designation outside of an intersection has two, a three-legged intersection has three, a four-legged has four, and so forth (with exceptions for distance breaks, which is explained later).

In addition to node IDs (NIDs), important fields in the usage table were "Knu_S_E," "Knu_Rt_Ne," and "Knu_Rt_Mp." The "Knu_S_E" field classified each node as S (start) or E (end). The letters occurred individually (e.g., it contained an S but not an E) if a route terminated at a node stack. But they were paired if it passed through a node stack (unless they were incorrectly plotted due to distance breaks). "Knu_Rt_Ne" contained the 17-character unique Route ID the node belonged to, and "Knu_Rt_Mp" was the milepoint along the route the node occurred on.

Using a companion shapefile called "AllRds_M" (which is short for: All Roads, measured), that is also maintained by KYTC, the node usage table was plotted using ArcGIS's linear referencing tools. Specifically, ArcGIS uses the route "Knu_Rt_Ne" and milepoint "Knu_Rt_MP" data to insert points at the appropriate distances along the line segments in "AllRds_M." This process can be expedited if the node usage table and the "AllRds_M" shapefile are first exported into a geodatabase (GDB). The resulting plot was a collection of points called an event layer, which we saved as a point feature class in the same GDB. Placing aside the problems introduced by distance breaks, we distinguished node stacks that constituted an intersection by counting the number of nodes in a stack (or nodes with the same ID number). A node stack with three or more nodes was either an intersection or part of an intersection. These were designated "centerline intersections" to distinguish them from the larger intersections they may be a part of.

Taken alone, centerline intersections do not adequately describe all intersections. Since they are based on lines of zero width, even a tiny offset between intersecting approaches results in two or more node stacks/centerline intersections that are clearly part of the same intersection (when the width of the actual roads are viewed). As shown in Figure 1, more complicated intersections often appear as multiple centerline intersections. For instance, a three-legged intersection involving a divided route (represented as

two parallel lines) results in two node stacks. A Y-intersection results in at least three, and interchanges or other very complex intersections can encompass many node stacks.

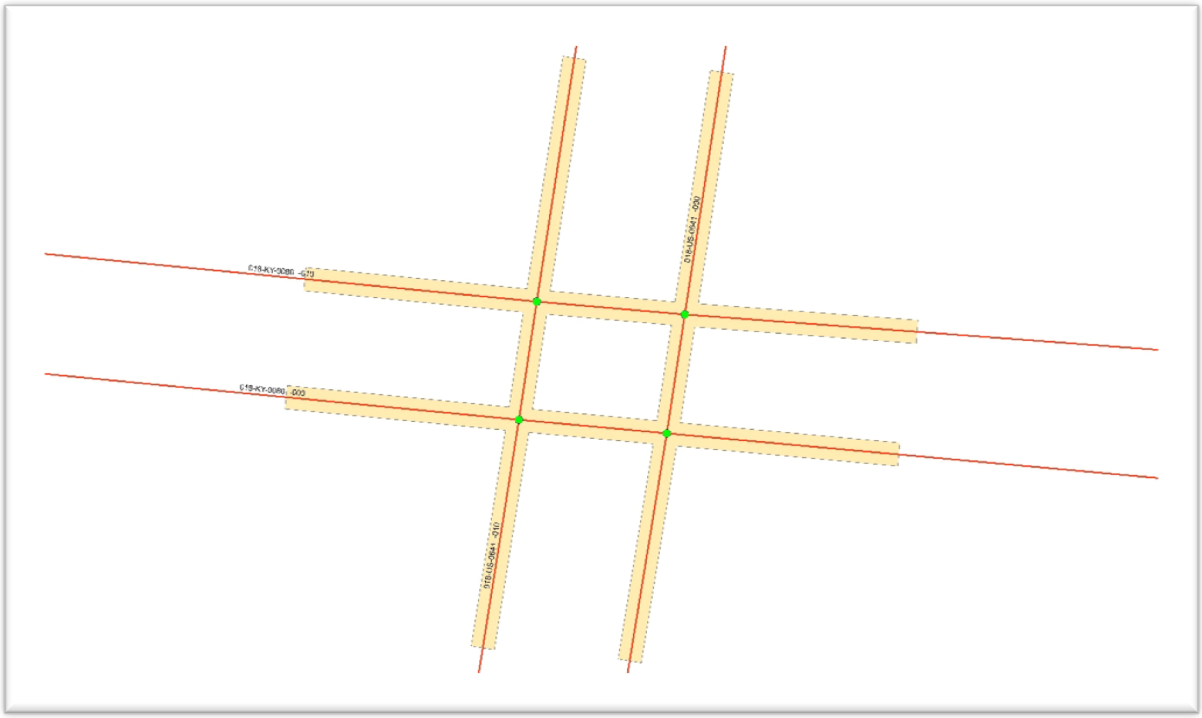

Figure 1: intersecting divided route, an example of one intersection containing four distinct centerline intersections.

2.1.2 Wandering Nodes

Distance breaks confuse the linear reference system and lead to node stacks that contain incongruent node IDs. Usually, routes can be represented as discrete and continuous lines, but in some cases breaks occur in the middle of a route, dividing a route into two or more discontinuous segments. Measuring the distance along the route (which linear referencing uses to plot the nodes) ignores the distance breaks, or treats them as wormholes, so that each discontinuous end of the break has an identical milepoint. Since linear referencing places only one point for each node, a point could randomly be placed on either side of the distance break. When that happens, the node usage table adds an extra node labelled as a distant break to mark it.

Although inserting a distance break node in a node stack throws off the 1:1 node-to-leg ratio, this is not a serious problem. Distance breaks are clearly identified in the table and can be selected and deleted. However, if ordinary nodes are displaced on the wrong side of a distant break, a more complicated fix is required.

Through a series of dissolves, summaries, and joins, the research team created an attribute table that contained all of the node stacks and the corresponding plot location of the multiple node IDs (NIDs). The attribute table included up to three fields for each NID; there was no stack that had more than three distinct NIDs. Assuming that all multi-NIDs were errors, we applied a series of tests to calculate which NID was the correct one for each stack. First, we assumed that whichever NID had fewer nodes in a stack was misplaced. Second, for cases where there were equal numbers of multiple NIDs in a stack, we checked to see if one of the NIDs had already been assigned – with certainty – to another stack. If it had we discarded it, and the remaining NID was adopted as the correct one. Third, for the remaining

undecided stacks, if one of the NIDs occurred exclusively in this stack and nowhere else, and the other(s) did not, the exclusive NID was taken to be correct. After performing the third test, a significant number of undecided stacks remained. However, none of these were located at intersections. As such, they could be ignored. In fact, all of the remaining set consisted of only two non-DB NIDs each: each connecting two dead-end distance breaks.

Using the information gleaned from this procedure, we created a corrected feature class of nodes. The wandering nodes were assigned to their proper coordinates – a feature class designated as "TrueNodes." There were a very small number of NIDs that legitimately occupied more than one coordinate, all of which were so close together (less than 12 feet) that there was no doubt that they belonged to the same intersection or were located on opposite sides of a divided highway. For the rest of "TrueNodes" records, each NID had exactly one spatial coordinate.

Using a dissolve let us count the number of nodes in a node stack and join the counts back to "TrueNodes" using the node ID. Those nodes that belonged to groups of three or more were then exported into a feature class called "IntsctNodes" (i.e., 'intersection nodes').

2.2 Tabular Processing

To move from node to intersection, we first plotted the legs of the approaches of the intersection (as lines). The node usage table already contained some of the information we needed, as each node record was defined as either the start (S) or end (E) point of a line segment. Each E node had a corresponding S node upstream with the same route ID. Except for non-cardinal routes, the sequence S-to-E followed the direction of increasing route milepoints in the state of Kentucky (milepoint information was embedded in "AllRds_M," which is why it was necessary to use this particular file for linear referencing). A portion of the route, usually about 100 feet outward from the intersection node, was defined as being within the zone of influence for an intersection (an area in which a given crash is likely to be influenced by proximity of the intersection).

Each node had a milepoint field. This was previously used to plot points with "AllRds_M." To create line segments for approaches, we attached to each node record another point that was a specified distance (100 feet by default) from it and along an approach. This segment's direction was defined by whether or not the node was an S or an E. For example, a four-legged intersection with no intersections nearby plotted as four 100-foot approach segments radiating out from a central point, which resembled an X or plus sign. Where intersections (i.e., centerline intersections) were close enough that their zones of influence overlapped, adjustments were made to the 100-foot distance. Adjustments also were made if the section number of the route indicated that the centerline intersection was part of a larger intersection system (such as an interchange or a Y-intersection). These adjustments took one of two forms. We either merged two or more nodes along the same route into one intersection or truncated their zones of influence to avoid overlap. An example of the merge logic is illustrated in Figure 2.

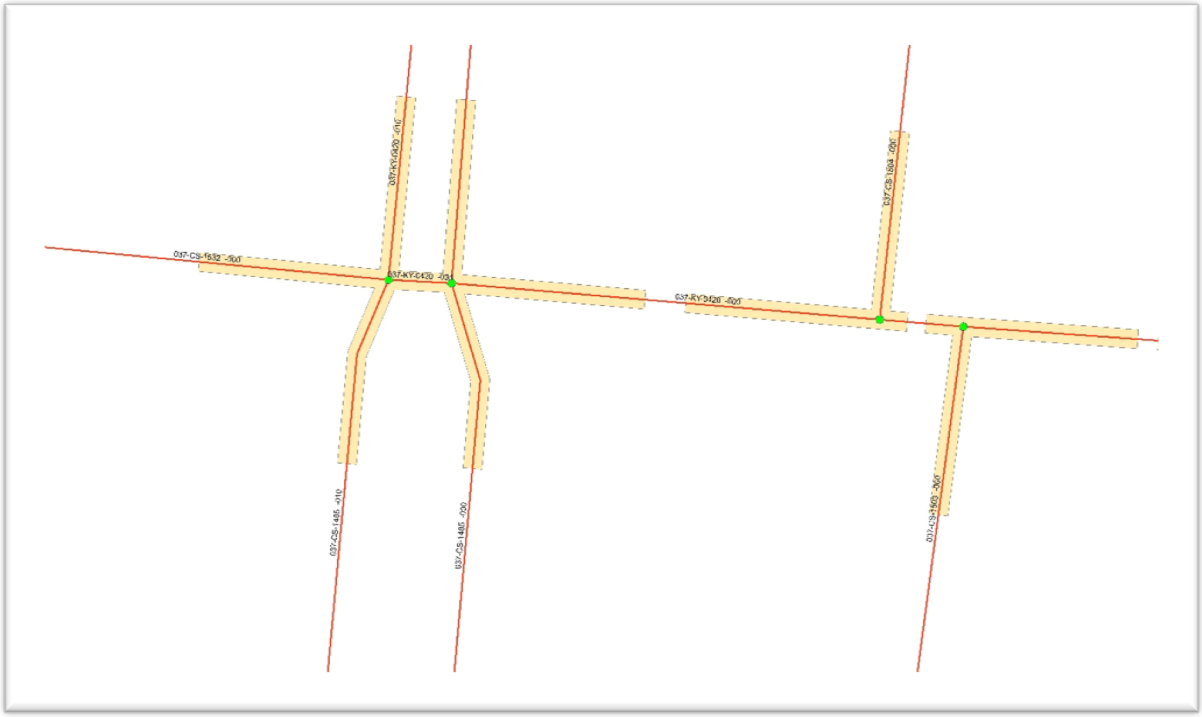

Figure 2: The process' merge logic merged the pair of node stacks on the left, but not the pair on the right.

We created the attribute table in Microsoft Excel because the software facilitated easy rearrangement of the data in the necessary sequence. Excel's cell reference system also expedited comparisons between adjacent records. Excel's major drawback was that it did not handle very large datasets well, which led to very long processing time and frequent crashes. We tried to limit this problem by deleting unnecessary nodes from the table. First, we deleted all distance break nodes. Then we deleted any node IDs (node stacks) with fewer than three of the same ID in a set (i.e., those that did not constitute a centerline intersection). This reduced the table to a more manageable size, although it led to another problem that stemmed from the absence of distance breaks, which is described at the end of this section.

The table was strictly ordered by route ID, S/E identifier, and ascending milepoints (the S/E identifier was identical to the original field, "Knu_S_E," except that it was reversed for non-cardinal routes). We developed formulas to compare each node record to the one before or after it (depending on whether the node was S or E) to determine whether the intermediate approach segments should be merged, truncated, or left at default. First we determined if two consecutive nodes were on the same route. If nodes along the same route were merged, we treated them as part of the same intersection. If two nodes' zones of influence overlapped but were not close enough to merge, their approach segment was truncated so that it equaled half the length of the gap minus 1/1000th of a mile (a small gap was left between segments to prevent their touching). The decision to merge or to truncate was based on the type of route, the type of route of the neighboring node, and the length of the gap separating them (i.e. length *along* the route, not the *absolute distance* between the two nodes). Any two node stacks with a gap less than 35 feet were merged into the same intersection. If the gap was more than 200 feet they were neither merged nor truncated. For gaps 35–200 feet long, nodes were defined as anti-merge, merge-happy, or neutral. Nodes belonging to routes with the section number 000 were classified as anti-merge and were never merged with one another at distances over 35 feet. Merge happy nodes were those that belonged to Y-intersections or non-cardinal routes, or had a neighboring node that did. These were merged for distances

under 200 feet. Anything else was considered neutral and was merged up to 100 feet, but they were truncated if the distance was between 100 and 200 feet.

In spatial terms, the product of the tabular calculations was the description of a line segment that either started or ended at each node. This segment could then be plotted with the linear referencing tool. For each node record, two fields were created to specify the milepoints at each end of the segment – one identical to the node's original milepoint, one at a distance (direction determined by the S/E identifier) that was ascertained through the merge/ truncate calculations described above. If we had performed a merge, the second milepoint was identical to that of the next node along the route.

In some cases, distance breaks threw off area-of-influence calculations. Calculating segment length took into account the position of the next node along the route and was truncated accordingly if another node was found within the specified distance. If the segment encountered a node stack with a distance break, it was cut off at that point. This would have worked for all distance breaks but we removed all node IDs with less than three real nodes (i.e. not including distance breaks), since these could not be part of an intersection. Excel was overtaxed performing this set of calculations on so many records. As such, we simplified the dataset as much as possible before doing the calculations. Where unmarked (non-intersection) distance breaks occurred within an intersection's area of influence, overshoots occurred. This produced detached line segments that appeared far removed from their associated intersections. Detecting and removing these during post-processing was not challenging, but in the future this step could be avoided by leaving non-intersection distance breaks in the table after flagging them for special treatment.

2.3 Intersection Database

2.3.1 Building Intersections in ArcMap

Plotting the modified table with "AllRds_M" using the Linear Referencing tool produced a large batch of line segments. When overlaid atop the previously plotted point features, these radiated outward from the point location of each node stack. In some cases this process linked nearby node stacks together. These segments were used to create polygons that represented each intersection. Because ArcMap is best equipped to combine touching or overlapping features as polygons, we created rectangular polygons based on each line with a buffer distance of five feet. We used the option for flat-ended buffers to preserve the gaps between proximate intersections. The dissolve tool was then used to merge all touching or overlapping rectangles into single polygons (this is why preserving the gaps between intersections is important).

The new polygon feature preserved the merge, truncate, and zone-of-influence logic from the tabular processing, but reduced the number of features/records to one per intersection. At this point, the new intersection features lacked attributes except for geometry and object ID (OID), which ArcMap adds to feature classes automatically. To give each intersection a unique identification, we copied the OID into a new field called "IntsctID" (OID itself is not a stable identifier because it changes when files are modified).

Creating the polygons resulted in several cases where intersection segments were merged inappropriately – mainly at overpasses. However, this set was small enough that errantly merged polygons were readily located (using the select-by-location function to find intersection polygons that overlapped another feature class, "PONTUS," which inventories Kentucky's bridges and overpasses). We manually edited the polygons to correct these mismatches. Several additional intersection polygons that represented pseudo-intersections (e.g., points at which a divided highway begins or ends) were later deleted. However, to detect those, we introduced additional data to the new intersection dataset. For this and other reasons,

attribute data from point and line feature classes were needed to populate the new intersections attribute table. We accomplished this using a process called stamping. To do this we executed a one-to-many spatial join of the intersection polygons to the points or line segments with the desired data. Doing this stamped each feature in the target with the intersection ID of the polygon it fell within. Desired attributes in the stamped features were then summarized (or dissolved with a "Statistics Field" set to sum by intersection ID). The resulting table was then joined back to intersection polygons with the newly derived data. To maintain clarity, we created two copies of the intersection polygon feature. One called "Stamps" was left as is. The other, called "Intersections," was augmented with attribute fields from later processing.

2.2.2 Derived Fields and Psuedo-intersections

To aggregate the data, we used the dissolve tool with statistical fields extensively on the node features to. First, the "IntsctNodes" feature was stamped with the intersection IDs so that each node falling inside one of the intersection polygons adopted the latter's unique identifier (the "IntsctID" field). "IntsctNodes" was then dissolved by various fields or by combinations of fields to count or otherwise summarize other attributes. For example, we counted the nodes in each stack by dissolving for Node ID with a statistics field that counted object IDs (or any other unique identifier). A secondary dissolve by intersection ID was carried out on the derived stack features in order to count the node stacks that fell within an intersection (in most cases there was only one stack per intersection, but more complicated intersections had several). We counted the number of unique route IDs in one node stack to identify loopbacks – points at which a route curves around to intersect itself. A loopback occurred if the same route ID appeared on 3–4 nodes in the same stack.

Counting the number of nodes with the same route ID within an intersection was crucial for calculating the number of approaches per intersection (see Figure 3 below). Any of these counts could then be copied into "Intersections" (the polygon feature) using a one-to-one table join.

For some of the desired statistical characteristics, additional fields were added to the feature class prior to the dissolve. Parts of the Unique Route ID ("Knu_Rt_Ne" in the original node usage table) were split off to create fields for Section ID (the last 3 characters) and LRS-ID (the first 13 characters). Section IDs were three-digit codes that defined routes based on their type. For instance, the Y-leg of a Y-intersection was defined by the range of section numbers, 020–029, and a crossover by 030–069. A section number of 000 indicated a standard route. A number of additional count fields were added to identify nodes by route types. These were then populated with ones or zeroes depending on whether the section ID fell within the range defined by that category. By definition, each node can only belong to one type. As such, nodes cannot have more than one of these count fields populated with a one. During the subsequent dissolves, a statistics field was added for each count. This field tabulated the number of nodes that belonged to each section ID category within the dissolve grouping.

Count fields helped researchers identify pseudo-intersections for deletion. Pseudo-intersections are: 1) divergence/ convergence points on divided highways, 2) isolated crossovers, 3) the divergence point of non-merging Y-intersections, and 4) route-congruent ramp intersections (i.e., intersection of ramps and other routes of the same LRS-ID). Node stacks with three nodes and a ratio of 2:1 between standard route (000) and non-cardinal divided highway route (010) nodes was defined as a divided highway pseudo-intersection. Any three-node stack that contained a crossover had the potential to be part of an isolated crossover. These node stacks were not deleted. They were used to define secondary binary count fields, this time with ones and zeroes in fields to signify different types of pseudo-intersections. Another dissolve of the node stacks by "IntsctID" grouped the node stacks in an intersection with counts of the number of node stacks that were pseudo-intersections. A new calculation field was then added to this table, which subtracted the number of pseudo-intersections from the count of total node stacks. If this field equaled zero (i.e., all of the node stacks in the intersection were pseudo-intersections), it flagged the entire intersection for deletion. Figure 3 below shows such an example of a deleted intersection.

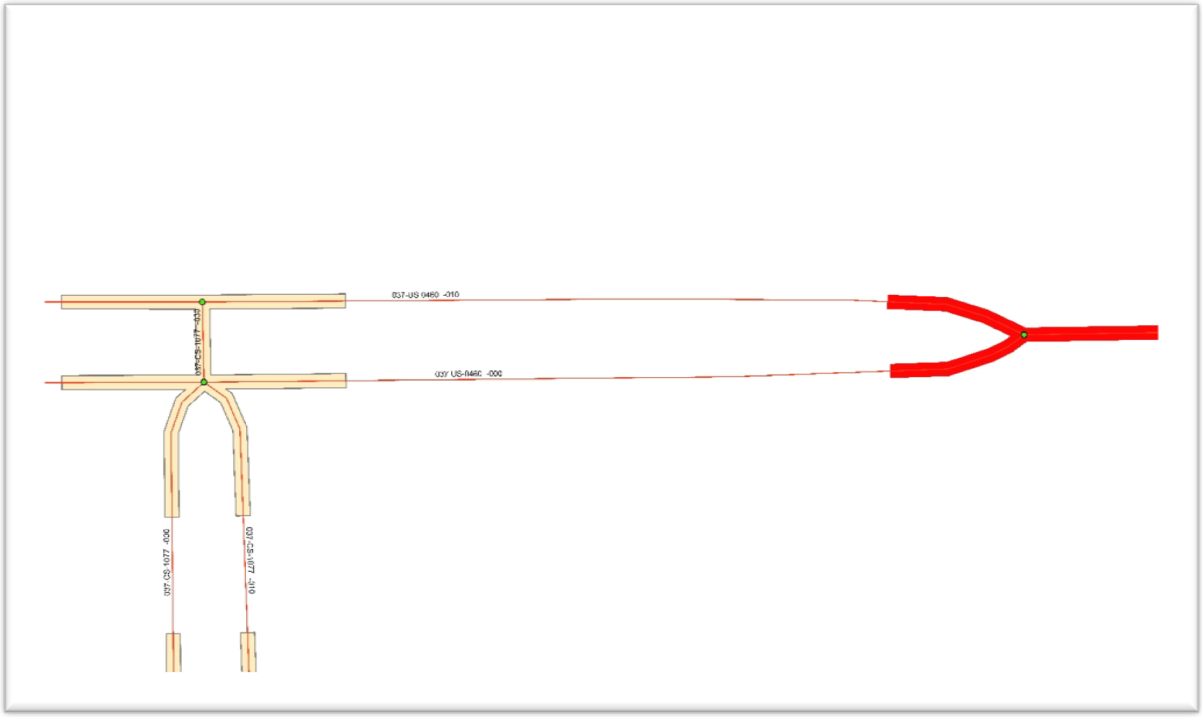

Figure 3: The intersection polygon on the right, highlighted in red, was deleted as a pseudo-intersection, representing merely the start of a divided route.

2.2.3 Route Ranking System and Master Nodes

We used route rankings to determine main routes and master nodes for each intersection. For this purpose, a table called "RouteRank," — containing all 17-character route IDs in Kentucky — was created. Each route was assigned a unique number, with lower numbers signifying a route that took higher precedence. The number used to rank routes contained up to 16 significant digits and extended out nine decimal places. The digits to the left of the decimal of the rank referred to the route's position within a hierarchy that was based on route prefix, government level, and route number. The decimal portion arbitrarily ranks routes that would otherwise receive identical ranks. It accounted for section IDs, county numbers, and route suffix (letter characters that sometimes appear at the end of route numbers).

The highest number component of the rank was the class modifier. This was based on a prefix and whether or not the route was a ramp. In some cases, it took into account government-level attributes. Ramps were assigned a class modifier of 2 million regardless of prefix. Interstates and parkways were assigned zero; U.S. routes were assigned 10,000; Kentucky routes 20,000; and XX routes (pending ownership designation) 30,000. All of these prefixes were defined as government level 1, while all other prefixes had a government level 2 or higher. These other prefixes were assigned a class modifier that was calculated by multiplying an integer by 10,000 (which was derived from adding 2 to the government level), so that the lowest government level would be 40,000; levels were spaced at intervals of 10,000 from there on. For example, a city maintained road has a government level of 4 would have a class modifier of 60,000.

Class modifiers (except those equal to zero) had four trailing zeroes so that route numbers – which at most had four digits – could be added to them without distorting the previously defined rankings. The class modifier plus the route numbers constituted the integer portion of the rank. The route suffix included in some route numbers (e.g., U.S. 31W) were removed from the whole number and included in the calculation of the decimal portion of the rank.

The decimal portions of the rank were derived similarly. Different components of the ranking were assigned three positions within nine decimal places. We limited numbers to nine decimal places because including a larger number bogged down ArcGIS GDB tables. The lowest component of the number (that is, the component furthest from the decimal point) was reserved for county number and was equal to county number (1 to 120) times 0.000000001. Section number (0 to 992) was assigned to the middle three decimal digits. It was calculated by multiplying the number by 0.000001. The first three digits after the decimal point were reserved for a number derived from the trailing letter characters of some route numbers. Because most route numbers did not include letters, these three digits were most often zeroes. Some trailing letters were followed by an additional trailing letter or a number up to 9. Both of these characters were converted to numbers and then multiplied by 0.01 and 0.001, respectively, and then summed. The first trailing character – always a letter – was converted to the numeric equivalent of its position in the alphabet (so, A = 0.01 and Z = 0.26). Converting the second character was complicated by the need to squeeze both numbers into three digits. To avoid different character combinations producing identical values, multiples of ten had to be avoided when converting the second character. If the second letter had been converted in the same manner as the first, so that J = 0.010, then the character set B and the character set AJ would both convert to 0.020. Letters were assigned numbers in alphabetical order, skipping 10 and 20, so that J = 11 (0.011 after the multiplication step), T = 22, and Z = 28. If the second character was the number one, we converted it to 29; all other numbers were converted by adding 29 to them (so, 7 = 36, or eventually 0.36). Thus, a trailing letter of WW would equal (23*0.01) + (25*0.001) = 0.23 + 0.025 = 0.255.

The entire ranking procedure we adopted is as follows:

> [class modifier] + [route number, stripped of trailing characters] + [trailing character 1 conversion]*0.01 + [trailing character 2 conversion]*0.001 + [section ID]*0.000001 + [county number]*0.000000001

This formula produced a unique number for any 17-character Unique Route ID. Any set of two or more routes (for instance in a node stack or intersection) can be compared to determine which one takes precedence.

By definition, intersections include multiple nodes (at least three, sometimes many more) and (except in the case of loopbacks) multiple routes. To anchor the intersection to one point, we selected one of the nodes at the intersection to serve as the index or master node. One of the routes at an intersection was also selected as the main route. One of the classification system's principal features (see below) hinged on this, namely whether or not the main route in the intersection was divided or undivided.

We determined both Main Route and Master Node using the "RouteRank" table. First a new field was created in "IntsctNodes" for Rank ("RtRank"). The route rank table was then joined to "IntsctNodes" based on route ID. Then, the rank from the table was copied into the new field and we removed the join. After this, we created a summary table for "IntsctID' with a statistic for minimum "RtRank." The summary table was then re-joined to "IntsctID'" based on the field "RtRank," by only keeping matching records. The matched records were exported to a new intermediate feature class, which was then dissolved by "IntsctID," route ID, node ID, and milepoint ("Knu_Rt_Mp"). The attribute table of the final dissolve contained a unique main route for each intersection. However, it sometimes included more than one candidate for Master Node. The latter situation arose in cases where the main route intersected nodes with different milepoints (those could be nodes with the same ID but different milepoints in the case of loopbacks). When this situation materialized, we assigned priority to the node with the lowest milepoint. Those were selected by repeating the summary-join-export procedure described above, but using minimum milepoint instead of minimum "RtRank."

The final table was joined to “Intersections,” and the new data were copied into newly created fields – denoted “MnRoute” and “MasterNd.”

2.2.4 Additional Data

Data for approach counts, main routes, and master nodes were all derived from information already contained in the node table. But additional data were required to classify intersections and to develop the safety performance functions (see below). These included data on traffic controls, traffic volume, directionality (one-way or two-way), crash statistics, jurisdictional level (state or local), and functional classes (including whether the route was designated urban or rural). Many more attributes were aggregated and integrated in the dataset, but these did contribute to the final version. Various methods were used to aggregate and integrate this data, but the most efficient method typically involved a variation of the stamping method described previously, using either “IntsctNodes” or “IntsctSeg” as intermediaries for the transfer.

The statistics for crashes for 2009–2014 (total crashes and crashes broken down by severity type) were based on a table that summarized the intersections in each segment according to route ID, beginning milepoints, and ending milepoints. Each crash was assigned a date, route, and milepoint. If a crash milepoint fell between the endpoints of a segment we assigned it to that segment, and count fields were generated for the total number of crashes as well as the number of crashes in each severity type (the KABCO scale was used). The resulting table was linear referenced, stamped with intersection IDs, aggregated by intersection ID with sum of total and subcategory crashes included, and joined with “Intersections.” This and any of the segment-based data (see below) could also be spatially joined to an already-stamped “IntsctSeg” feature class, and summarized by intersection later as one large batch.

AADT and other classificatory data were downloaded from KYTC’s website, drawn either from the fields of the “AllRds” shapefile or from more specialized datasets listed under Highway Information System (HIS). Data were acquired as shapefiles, however, they were not used directly. Instead, DBF tables were extracted and their data re-plotted using the standard version of “AllRds_M.” This step eliminated minor geometric differences found among some routes in the different datasets. These differences occurred because the Cabinet frequently updated its data. Before we re-plotted data, some tables were superimposed using the Route Overlay tool-- based on route and start and end milepoints. The Route Overlay tool superimposed segments defined by each milepoint pair and split any that overlapped into smaller segments. This preserved the attributes of all parent segments.

Re-plotted segments were clipped to intersections and stamped with intersection IDs. To determine if multiple new data segments overlapped one approach segment, we executed a select-for-location for segments that touched point features in “IntsctNodes.” If we had found any that did not touch an intersection node, they would have been deleted based on the assumption that the part of a route segment closest to the intersection took priority.

Where categories were very simple, researchers took a more straightforward approach of selecting-by-location and using the field calculator rather than directly joining features or attribute tables. For example, the “Jurisdiction” field in “Intersections” was populated by selecting state routes on a route plot, selecting-by-location intersections that touched the selected routes, and entering “state-state” in the field calculator for the selected features. A new selection for local roads was then created in the route feature class and another select-by-location done for “Intersections”, but this time with “select from current selection” checked. The result was a smaller selection all mislabeled “state-state.” Recalculating the selected features as “state-local” overwrote this smaller set while leaving the correct “state-state” category untouched. By default, all of the remaining unfilled records in jurisdiction were “local-local.”

The data for intersection controls was derived from a separate KYTC file called route logs. This shared a similar architecture as the node usage file, with points stacked at intersections. However, intersections were allocated one point per route, rather than one per approach. It also encompassed a much smaller subset of intersections than node usage. Like the node usage table, when plotted in ArcMap, points were situated on the wrong side of distant breaks. These were referred to as "junction-disjunctions" in the route log table. The research team used a procedure similar to the one described in the section 2.1.2 above to fix this. Not every intersection control point lined up with the exact point of an intersection in "AllRds_M," although most did. Non-aligning points were discarded, but most of these could be salvaged in future iterations of this database by using one or more of the proximity tools in ArcGIS to assign them to the closest intersections. The study used table and spatial joints to integrate the points that lined up with intersections.

The primary information used from route logs came from the field "CntrType," which classified traffic controls into a small number of categories including stop signs, traffic lights (signals), and no control. Since route log point features are specific to routes and not whole approaches, we assumed that where a route constituted two legs the traffic control type applied to both. Route logs were only available for a limited number of intersections, but often not for all of the approaches. We stretched this number by making a couple of assumptions. For signalized intersections, we assumed that control applied to all legs entering the intersection (whether or not the intersecting route was covered by route logs or not). Where the route log listed "no control" for a state route intersecting a local route and had no data for the local route, we assumed that the local route must have a stop control.

2.2.5 Approach Counts

Approach refers to a route through which traffic enters or leaves an intersection. For simple intersections with one node stack, the number of approaches equals the number of records (omitting distance breaks) with the same node ID. If intersections have multiple node stacks, counting approaches is more complicated. For example, a Y-intersection with three node stacks has nine individual nodes, but only three approaches. The segments between the node stacks that make up the intersection are internal to the intersection. As such, they do not constitute approaches. There are far more complicated combinations of node stacks that occur in some intersections, often combining multiple Y-intersections, divided routes, crossovers, and other types of converging routes.

To count the approaches in an intersection, the team first calculated the number of approaches per route within an intersection. Certain types of routes were considered redundant; that is, they were not separate approaches but merely a sub-component of one that had already been accounted for. For example, the cardinal and non-cardinal sides of a divided highway were classified as one approach. We resolved this by labelling NC routes as having zero approaches. We used the same logic for Y-intersections and crossover routes, since they were internal to the intersection and did not generate traffic that entered or left that had not been accounted for previously. For all other route types, we classified a route as having one approach if it terminated at the intersection, two approaches if it passed through the intersection, and three approaches if it formed a loopback. At a one-stack intersection, this equaled the number of nodes per route, but at complicated intersections a route might pass though several node stacks, generating two extra nodes for each one it passed though. See Figure 4 below for an example. A route, however, generated only one node on a route where it terminated (i.e., its starting or ending point). Therefore, a route with an odd number of nodes terminated at the intersection, where one with an even number of nodes passed through it (correcting for loopbacks).

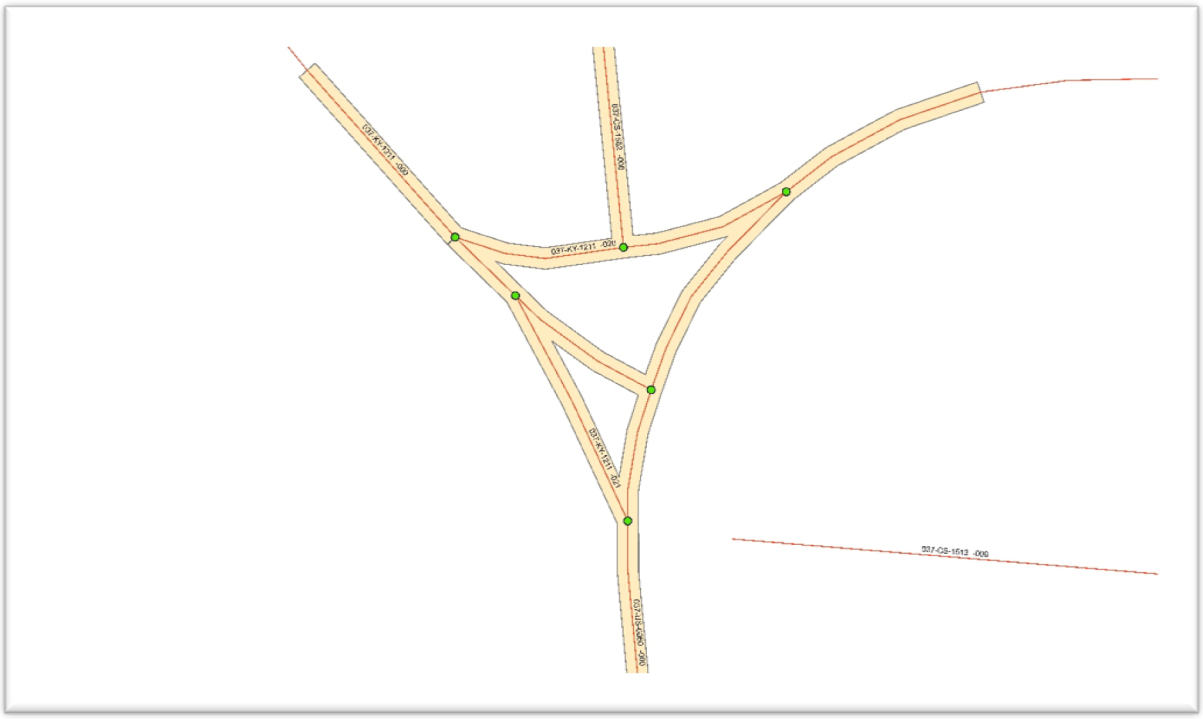

Figure 4: Intersection with four approaches, one each for KY 1211 and CS 1562 (odd numbers of nodes) and two for US 60 (even number of nodes).

To count the number of approaches per route, we used the new fields and dissolve-count statistics described in section 2.2.2. These calculations performed a series of joins and dissolves, which produced a table that counted the number of nodes per route per intersection, with additional fields indicating the presence of loopbacks and identifying redundant routes (any within the range 010–069). A field called "ApprRt" was added to this table indicate the number of approaches for each route. For loopbacks, this equaled 3, and for redundant route types, 0. For all other cases, routes were assigned a 1 if it had an odd number of nodes and a 2 was assigned for an even number. A final summary table was then generated for the "IntsctID" field with a field that summed all "ApprRt" numbers for that intersection. We then jointed this table with the Intersections feature class; the sum of "ApprRts" was copied into a field labeled "ApprCnt."

2.2.6 Calculating Entering Traffic

Researchers calculated the volume of traffic entering intersections by using the value of Average Annual Daily Traffic (AADT) for each leg, basing the estimate on whether it was a one- or two-direction leg. The AADT value was obtained from a downloadable HIS shapefile called "TF" (i.e., traffic flow); specifically, this information was contained in a field called "Last_Count." This was available for a limited number of road segments. It contained estimates of road traffic passing through the segment (both ways, except in the case of one-way routes). The Cabinet's "AllRds" shapefile contained the field "TYPE_OP," which defined whether a route was one-way, two-way, or part of a divided route. To correct for minor differences between "TF" and "AllRds" and the version of "AllRds_M" we worked with the DBF tables extracted from the "TF" file, combined them using the route overlay tool, and then re-plotted them with our "AllRds_M."

A spatial join assigned data from the new, combined event layer to the segments of the Approaches feature class wherever they overlapped. The "TF" data did not cover all of the intersection segments. For

intersections with missing data, a proxy value of 300 was assigned. 300 is an arbitrary number, one that is adequate for the pilot project. In future iterations of this database, we hope to improve estimates of the proxy values, possibly based on the AADT of nearby routes or with county-specific estimates.

Our goal was to estimate the total amount of traffic entering intersections, while the AADT ("Last_Count") values from "TF" included traffic in both directions (with the exception of one-ways). Therefore, AADT for two-way segments (or divided segments) were halved, while those for one-ways were either taken in their entirety or reduced to zero. This was contingent on what direction the traffic moved relative to the intersection. The following formulas were used for this calculation:

If TYPE_OP = 1 and Knu_S_E = S, entering traffic = LAST_COUNT*1
If TYPE_OP = 1 and Knu_S_E = E, entering traffic = LAST_COUNT*0
If TYPE_OP = 2 or TYPE_OP = D, entering traffic = LAST_COUNT*0.5

The formulas adjusted AADT figures for each route so they could then be summed together by intersection to gauge how much traffic entered intersections. For the purposes of the Safety Performance Function (SPF), however, this figure was split into two numbers – one for traffic entering via the intersection's main route (called "Major AADT" in the SPF) and one for traffic entering the intersection from all other routes at the intersection (called "Minor AADT"). In the rare event that either of these equaled zero (which could happen if the "Last_Count" field in the original traffic function table was zero, or if the only major or minor route was one-leg one-way leaving the intersection), we changed the number to 1 because the SPF development could not include zeroes in the major or minor AADT fields.

3. INTERSECTION DATBASE AND SAFETY PERFORMANCE FUNCTIONS

3.1 Intersection Classification

To develop Safety Performance Functions (SPF), the intersection dataset was divided into 20 categories. Approximately one-third of all intersections had sufficient attribute data to be classified using this scheme. Most of the excluded intersections were local routes that intersected other local routes. Typically, these lacked route-log information or AADT values. Table 1 lists the 20 categories, briefly describes them, and provides a count of how many intersections fell into each category. The category with the most intersections was "U3rP," which was undivided 3-legged rural intersections with a stop control on at least one leg. Other intersection types, such as "D3rS" are comparatively rare, and there were only a handful of intersections (for which data were available) that met this description in the entire state. There were a total of 182,384 intersections inclusive of all categories.

Table 1: Intersection Classification

Type	Code	Description	Count
1	U3rF	undivided 3-leg rural full stop	78
2	U3rP	undivided 3-leg rural (at least) partial stop	37256
3	U3rS	undivided 3-leg rural signal	96
4	U3uF	undivided 3-leg urban full stop	68
5	U3uP	undivided 3-leg urban (at least) partial stop	10252
6	U3uS	undivided 3-leg urban signal	583
7	U4rF	undivided 4+ leg rural full stop	77
8	U4rP	undivided 4+ leg rural (at least) partial stop	4202
9	U4rS	undivided 4+ leg rural signal	166
10	U4uF	undivided 4+ leg urban full stop	89
11	U4uP	undivided 4+ leg urban (at least) partial stop	2484
12	U4uS	undivided 4+ leg urban signal	1492
13	D3rP	divided 3-leg rural (at least) partial stop	729
14	D3rS	divided 3-leg rural signal	26
15	D3uP	divided 3-leg urban(at least) partial stop	1292
16	D3uS	divided 3-leg urban signal	335
17	D4rP	divided 4+ leg rural (at least) partial stop	459
18	D4rS	divided 4+ leg rural signal	66
19	D4uP	divided 4+ leg urban(at least) partial stop	560
20	D4uS	divided 4+ leg urban signal	832

Numerous features were examined before we settled on these categories. The classifications were based on three considerations: 1) whether the intersection was divided or undivided; 2) the number of approaches; and 3) the type of traffic control (e.g., full stop, at least partial stop, or signal). These variables existed in 24 combinations; four combinations (all of which were full-stop intersections on divided routes) did not occur in the dataset. This left the 20 remaining classes listed in Table 1.

3.2 Safety Performance Functions

We developed SPFs for each of the 20 intersection categories. For a given category, each intersection, its five-year crash history, and the AADT of each approach were compiled in a table. Using the R statistical software program, we performed negative binomial regression analysis twice for each category. The first analysis examined all crashes during the five-year period, while the second looked at only fatal and serious injury crashes (KAB). Regression analysis was performed on the data using the following functional form:

$$y = e^{a} AADT_{maj}^{b^1} AADT_{min}^{b^2}$$

Where: y = predicted number of crashes

$AADT_{maj}$ =Major approach entering average annual daily traffic

$AADT_{min}$ =Minor approach entering average annual daily traffic

a, b^1, b^2 are parameters from the negative binomial regression

The following code was used to generate the needed parameters:

```
#To read the data into R
data=read.csv("//C:/RevSPF.csv",header=T)
attach(data)

#Use this to specify a subset of data (comment out this and the next line if uneeded)
data<-subset(data, ClassNum==3)
attach(data)

#Define variables (comment and uncomment based on severity)
crash=(All_Crashes)
#crash=(KAB_Crashes)
logMajADT=log(MajAADT)
logMinADT=log(MinAADT)

#To load the MASS library which contains the command glm.nb
library(MASS)

#To generate the negative binomial models
init.theta = 3
model1=glm.nb(crash~logMajADT+logMinADT,init.theta=init.theta) ##Fitting

#To retrieve the summary of the models
summary(model1)
```

Table 2 summarizes the output of the regression equation for each intersection type.

Table 2: SPF Parameters

All Crashes				
Type	Intercept	Log_Maj	Log_min	Theta
1	-6.8508	0.7453	0.2635	0.86
2	-10.0896	0.84818	0.58105	0.8584
3	-6.78382	0.89225	0.15249	1.95
4	-2.40847	0.23563	0.25905	2.254
5	-6.58266	0.70294	0.29223	0.9444
6	0.13917	0.23507	0.10347	2.013
7	-8.17309	1.11404	0.08278	2.74
8	-9.69662	0.74447	0.72347	1.0202
9	-1.19747	0.20673	0.25183	1.895
10	-2.17487	0.19232	0.35288	1.363
11	-1.72875	0.2612	0.21699	0.9774
12	1.26001	0.11229	0.14798	1.6518
13	-4.97683	0.24124	0.59428	0.7245
14	-2.6982	0.4885	0.1142	1.945
15	1.61303	-0.06379	0.15371	0.6737
16	0.5548	0.20884	0.11222	1.948
17	-6.38656	0.25745	0.79386	0.891
18	-1.21364	0.18135	0.32262	3.57
19	-1.90987	0.26754	0.2528	1.0849
20	1.16855	0.15029	0.15579	1.8189

KAB Crashes				
Type	Intercept	Log_Maj	Log_min	Theta
1	-7.52232	0.74916	0.07044	0.225
2	-11.8225	0.77761	0.63695	0.6706
3	-8.29651	0.92564	-0.06765	1
4	-5.8866	0.1858	0.3356	4
5	-10.5728	0.8795	0.28738	0.9064
6	-3.22177	0.29948	0.11612	1.466
7	-14.6596	1.75784	-0.04074	0.51
8	-12.541	0.68717	0.93473	0.755
9	-9.8758	0.6791	0.5097	1.834
10	-2.24074	-0.06297	0.24041	0.676
11	-6.95671	0.51766	0.30547	0.7413
12	-1.6062	0.14811	0.14768	1.234
13	-8.8937	0.4526	0.6505	0.498
14	-4.6441	0.4695	0.1449	1.43
15	-3.51331	0.19135	0.20701	0.5931
16	-2.54404	0.26389	0.12154	2.538
17	-11.1975	0.5557	0.8629	0.577
18	-2.52751	0.15942	0.22736	1.872
19	-7.80787	0.57463	0.36916	1.348
20	-0.97881	0.12115	0.13871	1.794

4. CONCLUSIONS

This study produced a comprehensive, fully updateable intersection database. Adopting the node usage table as the database's foundation ensured that the location information in the intersection database was able to reflect any updates in roadway geometry that may have caused adjustments in downstream milepoints. The database will be revised regularly to reflect any edited, removed, or added nodes in the node usage table. Each intersection ID was indexed to one or more node IDs. Comparing the current node usage table to previous versions will let users identify any changes, flag them, and apply the appropriate correction to the associated intersection.

Using the SPFs developed as part of this study will provide a more data-driven approach to decision making on intersection safety, as compared to the previous critical rate analysis. We anticipate that considering the effects of regression-to-the-mean and ensuring that low-volume intersections are not ranked artificially high will assist KYTC in deciding where to prioritize the installation of intersection safety countermeasures.

There are many ways in which the processes described in this report could be refined and expanded in future versions of the database. A major limitation with the current version is the absence of AADT data and traffic control information for many (in fact, most) of the intersections in the larger database. It has been suggested that existing county-specific data for Vehicle Miles Travelled (VTM) could serve as a proxy to estimate AADT when exact measures are unavailable. Future iterations of the process will likely integrate interchanges as an analytical category. Preliminary work suggested that the skew angles between intersection approaches could be meaningfully integrated into the intersection classifications. Furthermore, it is anticipated that once the database and associated SPF results are made available to expert users, additional improvements will be suggested. As such, a quality control protocol will be implemented to ensure that errors in the database are addressed each year.

REFERENCES

Green, Eric, & Agent, Kenneth. (2003, August). *Crash Rates at Intersections*. Kentucky Transportation Center, University of Kentucky College of Engineering, Lexington, KY. Retrieved from Web: 10 September, 2014.

Kentucky Transportation Center. (2012, October 1). *Kentucky Strategic Highway Safety Plan 2011-2014*. KTC, University of Kentucky College of Engineering, Lexington, KY. Published By: Kentucky Transportation Cabinet Office of Highway Safety, Frankfort, KY. Retrieved from Web: 1 Nov. 2014.

Kentucky Transportation Center. (2013). *Kentucky Traffic Collision Facts*. KTC, University of Kentucky College of Engineering, Lexington, KY. Retrieved from Web: 3 Oct. 2014.

Srinivasan, Raghavan, & Carter, Daniel. (2011, December). *Development of Safety Performance Functions for North Carolina.* University of North Carolina Highway Safety Research Center. Chapel Hill, NC.

Simandl, Jenna K, Graettinger, Andrew J., Smith, Randy K., & Barnett, Timothy E. (2015). GIS-Based Non-Signalized Intersection Data Inventory Tool to Improve Traffic Safety. *Transportation Research Board Annual Meeting 2015 Paper #15-1585.*

Campbell, John R. & Knapp, Keith K. (2005). Geometric Categories as Intersection Safety Evaluation Tools. *Proceedings of the 2005 Mid-Continent Transportation Research Symposium*.

Garber, Nicholas J., Rivera, Griselle, & Lim, In-Kyu. Safety Performance Functions for Intersections in Virginia. *TRB 90th Annual Meeting Compendium of Papers*.