Edge of Pavement Extraction Using Image Processing Techniques

Jacob T. Lougee

IESA, University of North Georgia

GISC 4360K

Dr. Huidae Cho

May 5, 2021

Abstract

Many of the most recent remote sensing, and image processing techniques for finding pavement and pavement edge, are based on very high resolution datasets that currently require some fieldwork to obtain, such as drone and terrestrial lidar collection. Some of the current literature is focused on applications related to traffic management and road maintenance such as pothole and crack detection, vehicular navigation, and the rest. Some literature slightly more relevant to this project explores road centerline extraction. This project, though, is an exploration into extracting the edge-of-pavement from ESRI's aerial imagery and digital terrain model raster, using some image processing techniques in ArcGIS Pro, in order to reduce cost and man-hours involved in building a road-edge dataset for users of ESRI software.

Introduction

Every day, many GIS analysts spend some amount of time digitizing edge of pavement. This is because the permitting process for construction projects, especially of underground utilities near or under pavement, typically requires it to be represented on map documents conveying the construction plans. Because different roadways are maintained separately at the city, county, and state level, each governing body will inevitably have different GIS data on record. Some offer the data for free but this is uncommon, some charge a fee to access the data, some outsource the collection and maintenance of GIS data to private entities, and some don't have the data at all. The decision to use both a digital terrain model and imagery was made because, while some roads are very flat and have no curb, the edge may still be visible in imagery as a sudden color change, and while many roads have overhanging trees, the curb or edge may be detectable as a slight jump in elevation in the digital terrain model.

Literature

In, Cardim et al (2018, p. 2), while discussing their topic of evaluating different methods for road extraction, the author stated that, "Although the scientific literature has many published studies presenting methodologies for road extraction based on the digital processing of images from remote sensing, unfortunately, most, to the knowledge of the authors, focused only on a specific type of road and quite often with a limited set of characteristics or conditions. In this sense, even though road extraction methodologies have attracted much attention from the scientific community, they have also been a challenge due to the difficulty reproducing the complexity of roads characteristics as a general model [8]. Thus, there is a need for more studies that could tend toward generalizations of those problems presented by the scientific literature." This demonstrates the need for an effective pavement-edge extraction algorithm, and the benefit of combining digital terrain model data and edge of pavement data for the analysis, instead of simply focusing on imagery alone.

Cardim et al (2018, p. 4) also mention, "The mathematical morphology to post-process the segmentation result was used by Wang and Shan [5]. They classified the images into four groups (linear, curvilinear, crossings and breakages), creating a different post-processing for

each road type." While that is beyond the scope of this project, the last step of the process proposed by the paper is meant to be the use of the autoconstrain tool in cadmap3d, and perhaps such an approach could improve the results of that step.

Much of the literature about this topic makes efforts to classify the existing techniques used for road extraction, usually as structural, spectral, or hybrid techniques. This project might be considered a hybrid technique because while it largely examines pixel values making it spectral it also it involves edge detection techniques and considers a digital terrain model. Some object oriented classification might also be worth further investigation in the future, as Wang et al (2016) explains, in order to extract roads we can take advantage of, "geometric, radiometric or photometric, topological, functional and texture features" it may be worth noting that one of my peers this semester created a tool which quantifies the textures in an image, which could be used as yet another method for identifying pavement as, for instance, was done by Alshehhi and Marpu (2017) and, Sghaier and Lepage (2016) (not actually cited here) according to Cardim et al.(2018 p. 6 Table 1)

In terms of extracting road edges, as opposed to road networks, which was the primary goal in some of the papers just discussed, some emerging techniques include the use of UAV's and of mobile lidar or video/imaging sensors mounted to vehicles, see Kumar et al.(2013) and Zhu and Gu o(2020), The weakness of these methods compared to the one proposed in this paper is of course that they require the planning, field work and man-hours to collect the data, but as Kumar et al.(2013 p. 45) point out "The road edge is a fundamental feature and its efficient extraction is a topic which has not been extensively explored by the research community"

Project statement and Objectives

The goal for this project was to test for a viable method to automate the digitization of road pavement edge in GIS, by applying some of the image processing techniques that we learned about in class to the ESRI digital terrain model and basemap imagery. It is especially hoped that these ubiquitous layers in the ESRI environment can be utilized to great effect with this process.

Materials and methods

This project was completed in ArcGIS Pro and made use of the DIP toolbox, available at Dr.Cho's GIThub site https://github.com/HuidaeCho/dip-toolbox. Figure 2 is the GIS model utilized to extract the road edges. Keep in mind that not all of the tools were functioning properly at the time, for example the "bitwise NOT" tool did not work as anticipated so the reclassify tool was used instead to accomplish what the "bitwise NOT" tool should have accomplished (i.e. Set pixels 255 =to 0 and 0 =to 255) and the reclassify tool was labeled "bitwise NOT" in the model. The model itself is somewhat complex and the model builder will not allow names of anything to be repeated but, significant efforts have been made to label the functions and layer names in the model (again, figure 2) in what is hopefully a logical enough manner for any readers in the future

to follow along. Some of the naming got a little bit complex near the end, for example, some layers themselves were named in a convention somewhat like "and_and_andNOT". These names are best explained by the accompanying PowerPoint (slides 10-12). In addition the map package will be included with this essay, attempts have been made to delete a majority of the unused steps in the geo-processing history but as a precaution to avoid accidentally deleting any of the actual steps used, some of them were left untouched. Care has been taken to arrange the layers on the map in the order that they were produced (most recent on top - oldest on bottom).

The data used was ESRI default basemap imagery, and Digital Terrain Model data from "https://elevation.arcgis.com/arcgis/services/WorldElevation/Terrain/ImageServer" This data was chosen mostly for its availability to ESRI software users because this model is meant to eventually serve as an easy-to-implement process (note that it does still require some refinement). For this test, the process can be divided into two parts. The first part was to produce four outputs. They are; a raster of all pixels that matched a threshold for road color and its inverse, a mask of all the pixels around water and green tree edges, edges extracted from the image, and edges extracted from the digital elevation model.

Figure 1 shows a suggested methodology for implementing this over a large project-area defined by a point, line, or polygon layer, as well as the steps that were taken to ensure that both of the initial layers were perfectly aligned, with a one-to-one pixel ratio.

The road color was extracted by applying a color slice. Road color pixels were given a maximum value and everything else, a value of zero. For later testing of different methods, the inverse of this raster was calculated using the "bitwise NOT" operation.

In order to get the pixels that represented the water and tree edge green colors, a color slice was utilized. The resulting image has water/Green pixels represented as a value of 255 and the rest as 0. Because, I knew that I wanted to eliminate these pixels from the edges, I reversed that using the "bitwise NOT" tool, however I realized during my presentation that this was an extra step which I had done before I knew that I was going to buffer these pixels. So, for anyone researching this in the future, the "bitwise NOT" is no longer necessary here. Those pixels were converted to polygons using the raster to polygon tool. Next a definition query (labeled "feature class to feature class" in figure 2) was applied to select only polygons with a gridcode of, in this case, zero. And they were exported as a new feature class. A buffer of two feet was applied to this feature class. The resulting polygon was converted back to a raster to represent a buffer around these features. The pixels in that raster were reclassified, and this is the step that makes the earlier "bitwise NOT" redundant, the pixels representing water/green are given a zero value, and everything else as the maximum value (255).

Extracting edges from the color image was essentially a three step process, it was first converted to grayscale and then the local statistics, standard deviation, was applied to a five by five neighborhood. A threshold graylevel slice was used to convert edge pixels to the maximum value, and non–edge pixels to zero.

To get edges on the digital terrain model, local statistics standard deviation was used again, however, because the values were not dramatically different, it was very difficult to

actually extract the edges and a twenty by twenty neighborhood was used. For anyone trying to improve this in the future, I would suggest some method of pre-processing like sharpening, and/or multiplying pixel values or raising them to a power first. Whatever it takes to seriously enhance the results of this edge extraction step would significantly improve the overall result.

Using those four outputs from the first part of this process, three methods were tested to extract the road edges. In method one, the image edges and terrain model edges were used as inputs in a "bitwise OR" operation This was compared to the non-water/non-tree–edge-green buffer pixels in a "bitwise AND" operation., and the output of that was compared to the with the road colored pixels in a "bitwise AND" operation. In summary, method one extracts pixels that are color edge or terrain edge, and not within two feet of water-green pixels, and road-like in color.

In method two, the image edges and terrain model edges were used as inputs in a "bitwise AND operation". This was compared to the non-water/non-tree–edge-green buffer pixels in a "bitwise AND" operation, and the output was compared to the with the road colored pixels in another a "bitwise AND" operation. In summary, method two extracts pixels that are color edge and terrain edge, and not within two feet of water-green pixels, and road-like in color.

In method three, the image edges and terrain model edges were again used as inputs in a "bitwise AND" operation. Which was compared to the non-water/non-tree-edge-green buffer pixels in a "bitwise AND" operation, and the output of that was compared with the inverse of the road colored pixels in a "bitwise AND" operation. In summary, method three extracts pixels that are color edge and terrain edge, and not within two feet of water-green pixels, and are not road-like in color. For all three methods, the selected pixels were then "thinned" using the "thin raster" tool, converted to polylines using the "convert to polylines" tool, and exported to CAD to be auto-constrained.

Some logical errors in this process were identified via the testing of these three methods so, in the simplest possible terms here is the process I would recommend:

Preprocess and get a higher quality edge extraction from the DTM (edges 255, else 0)
name: DTM-edge

Do the same edge extraction/grey-level-slice from the imagery that I used. (edges 255, else 0)
name: I-edge

Color-slice tree edges and buffer as I buffered (buffer around tree edges 0, else 255)
name: notT-edge

Color-slice water and buffer it (buffer around water 0, else 255)
name: notW-edge

Color-slice road then "bitwise not" like I did (road color 0, else 255)
name: notRoad

Now, run these comparisons:

DTM-edge AND notW-edge (name: DTM-edge2)

I-edge AND notW-edge (name: I-edge2)

I-edge2 AND notT-edge (name: I-edge3)

I-edge3 OR DTM-edge (name: Combined-Edges)

Combined edges AND notRoad (name: Pavement-Edge)

Thin---->Convert to Polylines---->Export to CAD as I did.

Auto Constrain

Results

The original plan, labeled method 1, was the most inclusive, and was the only method that captured the edge of the parking lot pavement in the Northeast corner of the image. Method three captured some stray pieces of the difficult-to-capture road in the southern part of the image All the methods did very well at removing water edge and tree edges, but removing tree edges from the edge of both models caused some significant errors of omission. If future researchers can improve the result of the edge extraction from the digital terrain model it would be better to first remove water/green edges from the imagery edges, then only water or a water buffer from the digital terrain model edges.

The next step in the process is meant to be the use of the Auto-constrain feature in CAD-map3D. The auto constrain tool asks for an ordered list of what constraints should be applied to line features, and for the threshold values. So for example, we could say that we want any lines that are within a few degrees angle of one another to be redrawn so that they are aligned parallel, any parallel lines that are within a few feet of each other to be redrawn so that they are aligned collinearly, and any lines whose ends are within a few feet of each other to be redrawn so that they're ends touch. This may very well have been able to fix some errors of omission, but the software license was not available.

Conclusions

It is clear from this testing that this process might require a much more precise digital terrain model, or much more pre-processing of the DTM used here, in order to work properly. Some alternatives worth exploring might be sharpening, multiplying each pixel value, or even raising them to a power or log, in order to exaggerate the features in the digital terrain model.

The results of the testing also suggest that a better approach might be to isolate water and green edges separately, and take green from image edges, before comparing them to the digital terrain model edges. A process similar to the one experimented with here could certainly be viable for extracting road edges, however more testing is required to perfect the process.

Acknowledgment

Thanks to Dr.Cho for creating and sharing the digital image processing toolbox, which was very useful for this research.

Appendix

Figure 1

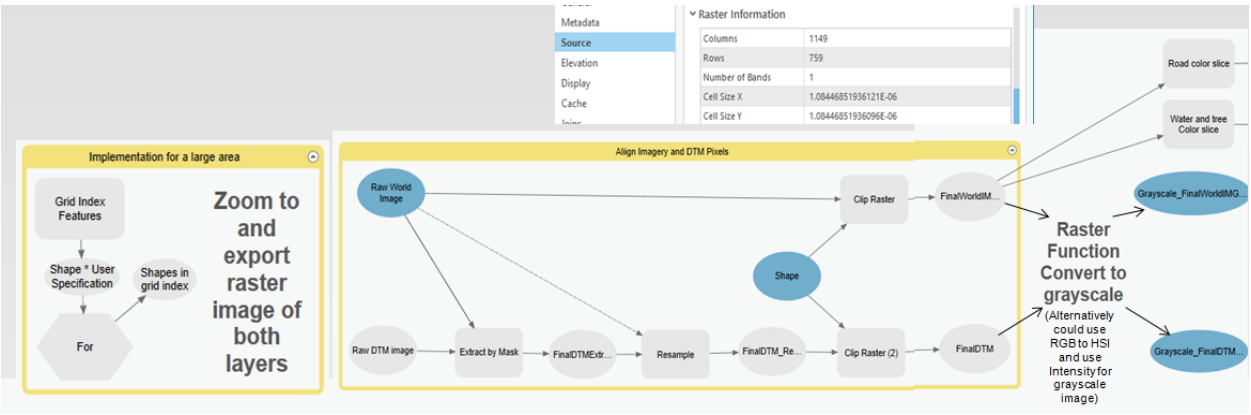


Figure 2

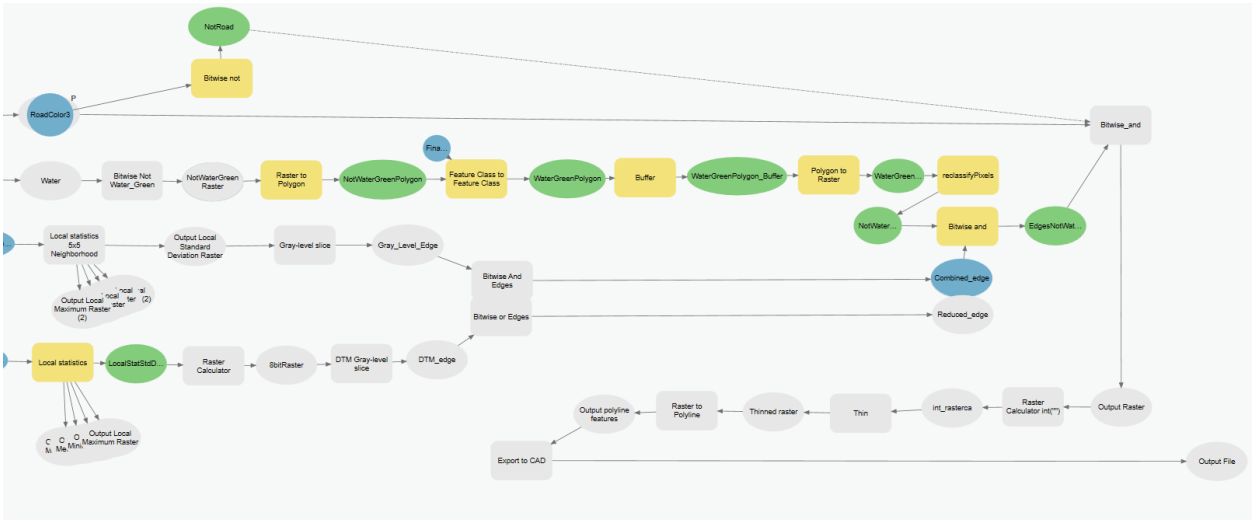


Figure 3

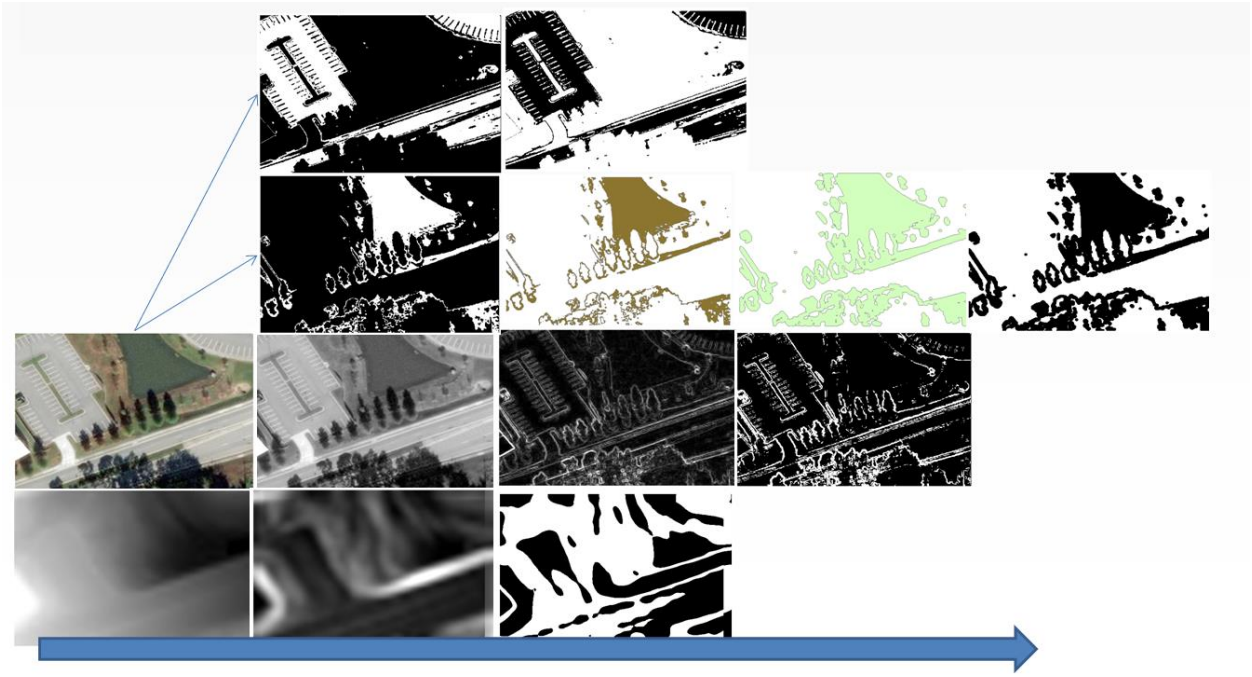


Figure 4

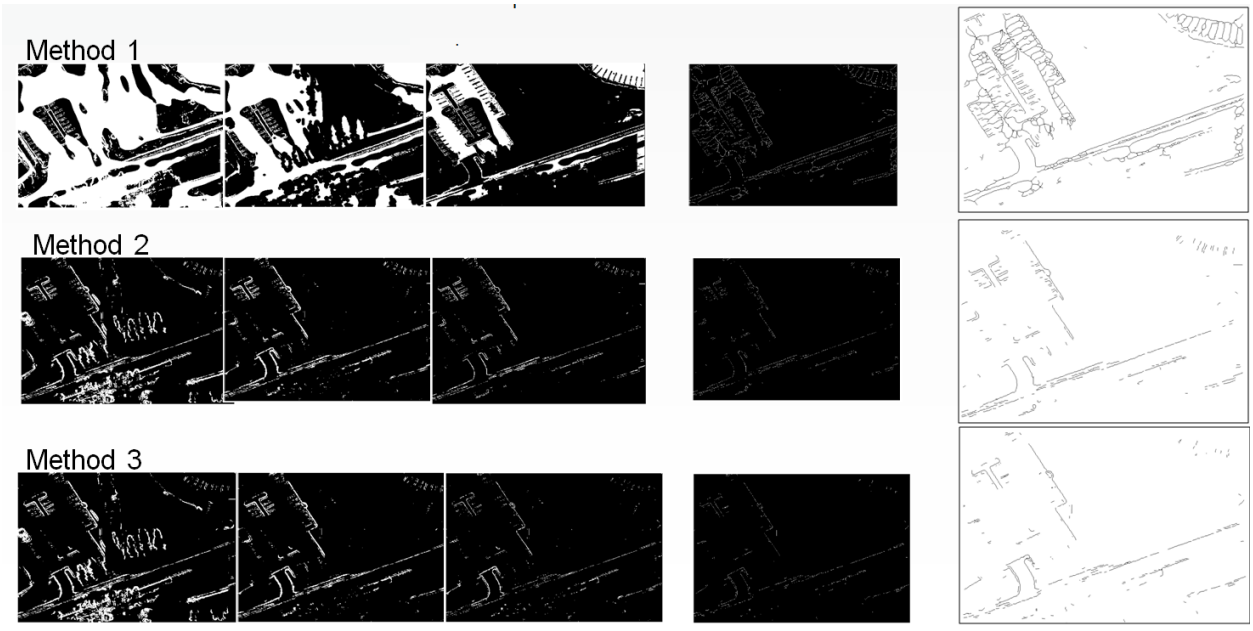


Figure 5 (method1)

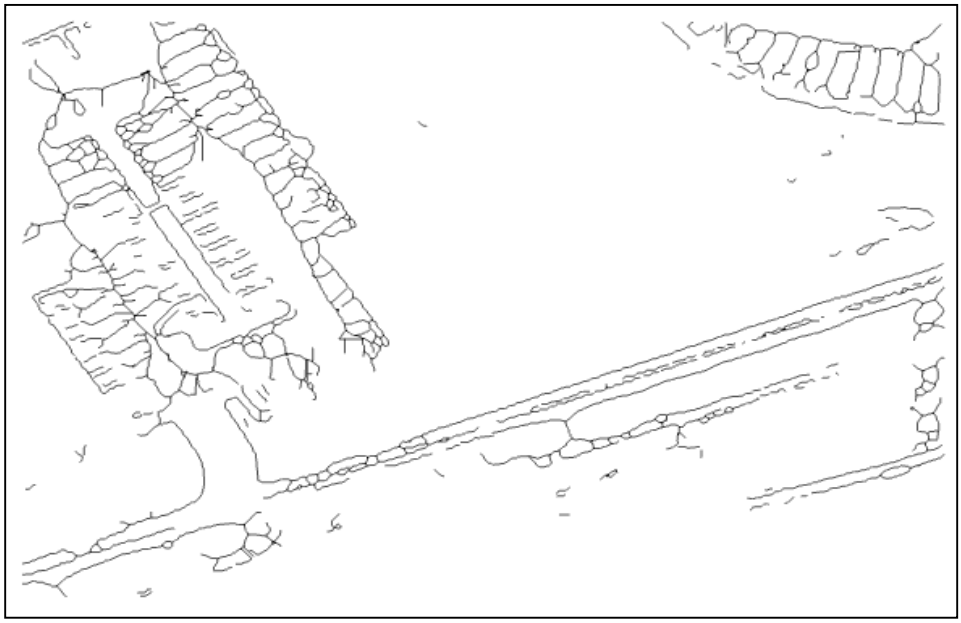
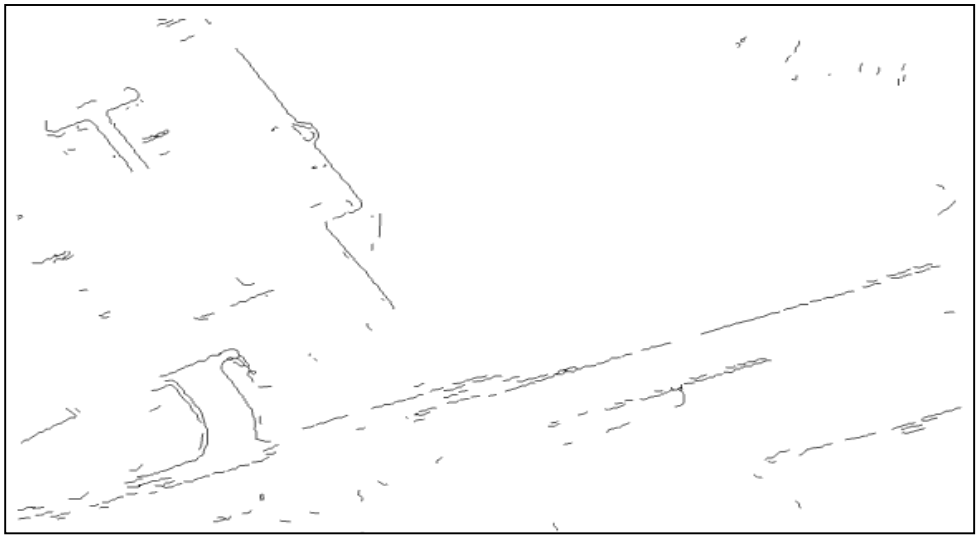


Figure 6 (Method 2)

Figure 7 (Method 3)



References

Ahmed, K., Al-Khateeb, B., & Mahmood, M. (2019). Application of chaos discrete particle swarm optimization algorithm on pavement maintenance scheduling problem. Cluster Computing, 22, 4647–4657. https://doi.org/10.1007/s10586-018-2239-3

Cardim GP, Silva EA, Dias MA, Bravo I, Gardel A (2018) Statistical evaluation and analysis of road extraction methodologies using a unique dataset from remote sensing. Remote Sens 10:620. https://doi.org/10.3390/rs10040620

Cardim, G.P., da Silva, E.A., Dias, M.A. et al. A nonrecursive GR algorithm to extract road networks in high-resolution images from remote sensing. Earth Sci Inform 13, 1187–1199 (2020). https://doi.org/10.1007/s12145-020-00501-5

Huang, H., Fan, R., Zhu, Y., Liu, M., & Pitas, I. (2019). A Robust Pavement Mapping System Based on Normal-Constrained Stereo Visual Odometry.

Kumar, P., McElhinney, C. P., Lewis, P., & McCarthy, T. (2013). An automated algorithm for extracting road edges from terrestrial mobile LiDAR data. ISPRS Journal of Photogrammetry and Remote Sensing, 85, 44–55. https://doi.org/10.1016/j.isprsjprs.2013.08.003

Liu B, Wu H, Wang Y, Liu W. Main Road Extraction from ZY-3 Grayscale Imagery Based on Directional Mathematical Morphology and VGI Prior Knowledge in Urban Areas. PloS one. 2015;10(9):e0138071. doi:10.1371/journal.pone.0138071

WangW, Yang N, Zhang Y,Wang F, Cao T, Eklund P (2016) A review of road extraction from remote sensing images. Journal of traffic and transportation engineering 3:271–282. https://doi.org/10.1016/j.jtte.2016.05.005

Wulamu, A., Shi, Z., Zhang, D., & He, Z. (2019). Multiscale Road Extraction in Remote Sensing Images. Computational Intelligence and Neuroscience, 2019, 2373798. https://doi.org/10.1155/2019/2373798

Yang, M., Liu, X., Jiang, K., Xu, J., Sheng, P., & Yang, D. (2019). Automatic Extraction of Structural and Non-Structural Road Edges from Mobile Laser Scanning Data. Sensors (Basel, Switzerland), 19(22). https://doi.org/10.3390/s19225030

Yiyi Zhu, & Na Guo. (2020). Unmanned Vehicle Route Tracking Method Based on Video Image Processing. Jordan Journal of Mechanical & Industrial Engineering, 14(1), 139–147.