Utilizing Sensing Capability Of Autonomous Vehicles To Improve Road Safety

Shiva Chandrashekar
ASU ID: 1215106256
snchand1@asu.edu

***Abstract*—The paper discusses the effectiveness of mobile devices in providing edge processing capabilities to autonomous vehicles. The paper also implements an extension to the coverage algorithm that utilizes mobile sensing.**

***Index Terms*—Sensing, Autonomous Vehicles, Road Safety**

I. Introduction

With the abundance of sensing capabilities, autonomous vehicles can be seen as a fully-functional unit. The need for a remote processing unit is eliminated with mobile devices acting as local processing unitsv providing reduced latency rates allowing the use of deep learning techniques for processing large image, signal and audio data-sets. A clear improvement to traditional methodology. But with the development of these vehicles, the need for road safety goes hand in hand. It can, for example, encounter a human-driven car, meaning it would be a harder task to maintain coordination.

Some of the applications of mobile edge analytics include Heterogeneous data sources, vehicular platoon control, Coverage control, and path planning. For the implementation, an extension to [7] is provided.

The key idea behind the coverage algorithm is to a region for each agent to be responsible for. This region is called the dominance region and would be a small portion of the whole area to be covered. The idea of the implementation is to enhance the algorithm to make it more adaptive to a situation where there could be an obstacle in the map where the agents are deployed. This adaptive capability will make the algorithm more robust in terms of the location of the agents. This extension is not discussed in the paper and would make it more suitable for the real world.

II. Project overview

Discusses some of the concepts discussed in the introduction and then delves into the implementation.

A. Heterogeneous data sources

The need for various types of sensing capabilities for autonomous vehicles has meant that data to be processed is highly heterogeneous in nature. For example, there might be sensors placed to find the distance to the nearest vehicle situated in the front or there could be one that tracks the driver's EEG or fatigue level. It is important to make sense of all the data that is provided. By providing edge computing capability, we can apply Multimodal learning with deep Boltzmann machines(RBM) on heterogeneous data which will result in better control decisions rather than handling each type of data separately. A Multimodal RBM helps in extracting a unified representation by joining multiple modalities together.

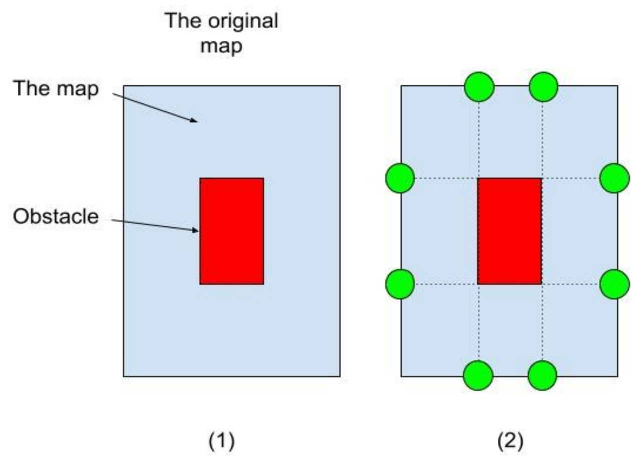

Fig. 1. The map and the obstacle

B. Path Planning and autonomous control

Essentially autonomous cars are expected to navigate in the most optimal way possible. With the current development of GPS, it is simple to find the optimal destination for the path. Combining that with the sensor measurements of the autonomous vehicle recorded dynamically, would make it more efficient. All the collected information can be processed on the edge with the help of an LSTM and CNN approach. The CNN processes the images to identify the objects with the help of features of the environment, while the LSTM decides the sequences of actions along with their paths.

C. Semi-autonomous ITS's

A major challenge in autonomous vehicles is the existence of human drivers in the midst of self-driving vehicles. This would make it a hazardous situation owing to the unpredictability of human drivers. Instead of deploying extra sensors to handle the situation, an alternative could be employing edge analytics to predict the behavior of the human. A self-driving car follows a human driver to identify the decisions taken by it at different scenarios and instances. By recording these actions and feeding them into an RNN, one an predict the actions taken by the human in the future. The accuracy of these predictions with more data being fed into the RNN.

D. Coverage control for mobile sensing networks

With the development of mobile robots and their ability to communicate with one another, they can be used as agents to provide uniform coverage over an area.

A Voronoi diagram is a partition of the plane into cells, where the partition is based on a set of points P, in the plane. For each cell directed by a particular point, the points inside the cell are closer to this point than any of the other points in p. Each Voronoi cell will be associated with a corresponding generator point.

Since we strive to provide the maximum coverage with the agents provided to us and the location optimization function is given by the euclidean distance between any point where we want to provide coverage and the nearest agent in the area, it is derived that the local minimum for the function is achieved by selecting the centroid of each Voronoi region as the position of the agents.

1) Algorithm: The algorithm can be briefly said to be 3 basic steps

1) Find the Voronoi partitions for the n agents
2) Find the centroid of these Voronoi partitions
3) Deploy the agents at the corresponding centroids

E. The extension

Our extension to the coverage control algorithm is by adding obstacles to the map. Both the obstacle and the original map had to be a convex polygon. The strategy is to divide the map into smaller convex sub-maps.

F. Finding the submaps

1) Finding the intersection vertices: The lines of the obstacle (2 adjacent vertices form a line) are extended to intersect with the boundary of the original map. Since the original map is a convex polygon, it is assumed that the lines would intersect only at 2 points. The theory is represented in 1. The green circles indicate the new intersection vertices.

2) Assigning obstacle vertices to map vertices: To form the sub-maps, the idea is to pick 2 consecutive vertices at a time (original vertices + new vertices from the intersection) and find the vertex of the obstacle closes to them and assign them to each other. The closest obstacle vertex becomes the current test vertex. The vertex that follows this vertex becomes the next test vertex. For each vertex from the map, we find which vertex (current test vertex, or next) is the closest. The goal of this code is to keep the order of assignment to guarantee that the resulting submaps are convex.

3) Creating the submaps: The final step to create the submaps is to create polygons. Each polygon has two map's vertices, and one or two of the obstacle's vertices. Our algorithm guarantees that all polygons are convex. Also, polygons with two map vertices and two obstacle vertices are convex because our algorithm avoids having a vertex inside the polygon that is not part of it.

G. Combine the submaps into bigger submaps

Once all the submaps are found, it is important to combine them to form the bigger map to run the coverage algorithm. Since we have to maintain the convex nature of the map, we start by picking the first submap from the submap list and combine it with the next, as long as the combined map is still convex we keep appending to it, if there is a situation that does not abide by the constraint, then we add the previously formed polygon to the result set and start with the current polygon afresh. This represented in Figure 2 where on the left the small 12 polygons were combined to 4 big submaps.

H. coverage control

Once we have the combined map, we implement the coverage control algorithm. The algorithm expects a polygon (the map) and the number of agents. The inputs to our extension are the map vertices, obstacle vertices, and the number of robots. The output is the coverage partitions. After getting the big submaps, we find the ratio of the area of the submap to the original map area, excluding the area of the obstacle. Then we assign the same ratio of robots to this submap.

3, 4,5,6 represent the number of agents being deployed in the 4 big submaps that were formed after the smaller maps were combined to a bigger map. The number of agents deployed was based on factors as discussed above.

III. Conclusion

With the advent of autonomous vehicles and the capability of mobile sensing and processing, it is possible to enhance road safety by incorporating deep learning techniques to solve real-world problems. With respect to the implementation, the strategy was to split the original map into multiple convex sub-maps, the belief is that our solution produces excellent results for the problem of coverage control in the presence of an obstacle, but we do not guarantee the optimal solution.

IV. Contribution to the project

I had to initially read a lot of research papers to understand the influence of mobile devices in providing processing capabilities. Once I had a clear idea of mobile edge analytics, I read more on its various applications. For the implementation, I was responsible for splitting the bigger sub-map into various smaller ones as discussed in the implementation section of the code. The bulk of the coding was done using python, with the help of various geometric functions.

V. Skills gained

Reading close to 15 scientific papers helped me understand the nuances of finding the essence of each paper and how to find the aspect of the paper most relevant to the topic at hand. I could follow most of the papers I picked for my project because they were extensions to the concepts taught in class. Once I had an idea of the theory, expressing my thoughts on paper taught me to be more concise in my explanation. For most of my coding, I used python, so it helped me improve on my existing knowledge of the language. Since the extension to

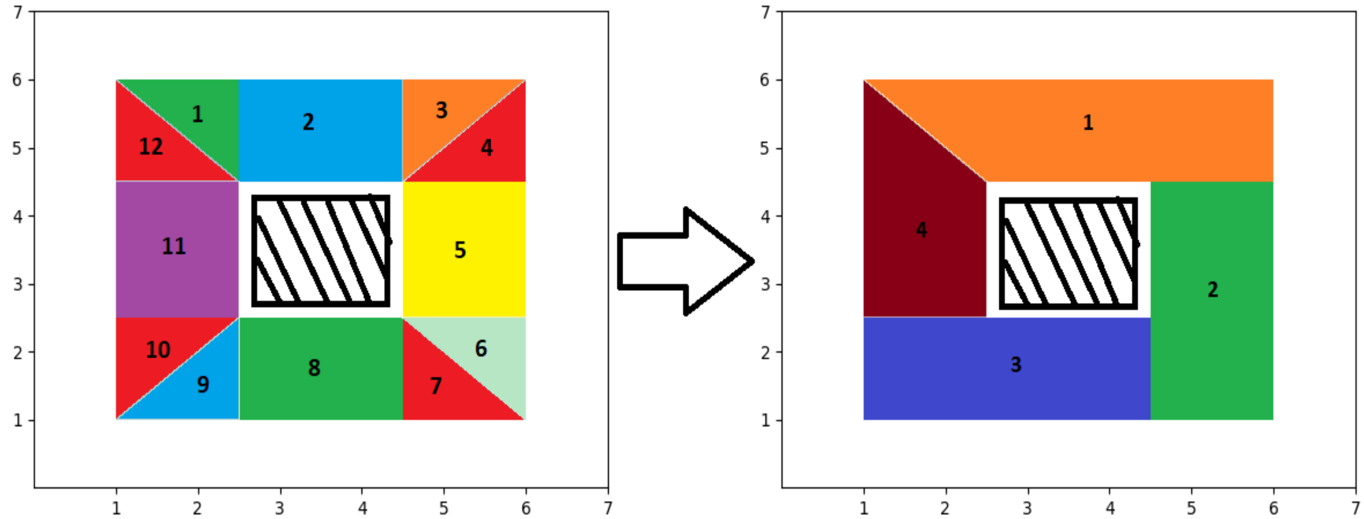

Fig. 2. Combine the submaps into bigger submaps

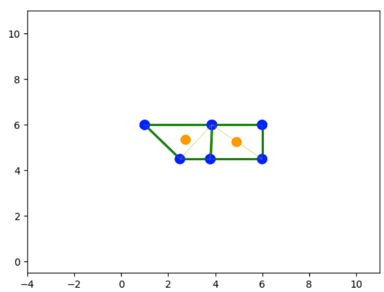

Fig. 3. coverage partitions for submap 1 from fig 2

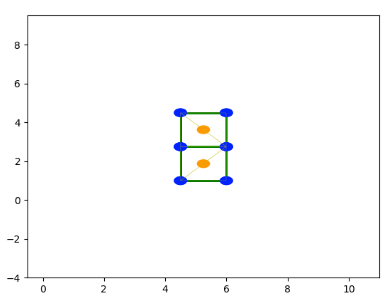

Fig. 4. coverage partitions for submap 2 from fig 2

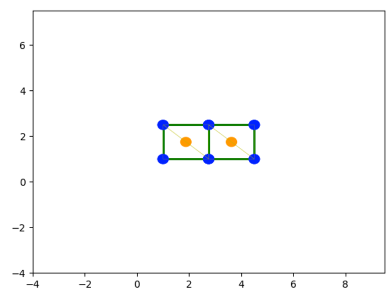

Fig. 5. coverage partitions for submap 3 from fig 2

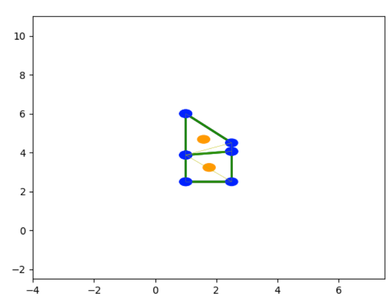

Fig. 6. coverage partitions for submap 4 from fig 2

the coverage paper was in-fact a real-world problem, it gave me more of an idea about the problems faced in an industry setting.

VI. ACKNOWLEDGMENTS

I would like to thank Dr. Stephanie Gil for allowing me to investigate such a problem and helping us every step of the way. I would also like to thank the TAs Ninad Jadhav, Sushmita Bhattacharya, and Weiying Wang. Special thanks to my team partner for his continued support.

- Team Members
  1) Faisal Alatawi

REFERENCES

[1] M. Gerla, E.-K. Lee, G. Pau, and U. Lee, “Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds,” 2014 IEEE World Forum on Internet of Things (WF-IoT), 2014.

[2] M. Gerla, “Vehicular cloud computing,” 2012 The 11th Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net), pp. 152–155, 2012.

[3] Papadimitratos, P., Fortelle, A. L., Evenssen, K., Brignolo, R., & Cosenza, S. (2009). Vehicular communication systems: Enabling technologies, applications, and future outlook on intelligent transportation. IEEE communications magazine, 47(11), 84-95.

[4] Intelligent transportation spaces: Vehicles, traffic ... (n.d.). Retrieved from https://bit.ly/33TQuTV.

[5] Ferdowsi, Ursula, Saad, Walid, & B., N. (2018, May 8). Robust Deep Reinforcement Learning for Security and Safety in Autonomous Vehicle Systems. Retrieved from https://arxiv.org/abs/1805.00983.

[6] Ferdowsi, Ursula, Saad, Walid (2017, December 12). Learning for Reliable Mobile Edge Analytics in Intelligent Transportation Systems. Retrieved from https://arxiv.org/pdf/1712.04135.pdf.

[7] Cortes, Jorge, et al. ”Coverage control for mobile sensing networks.” IEEE Transactions on robotics and Automation 20.2 (2004): 243-255.