Name: Yunchu Feng
Name of Project Advisor(s): Dr. Patricio Vela
Group Name: Team Satellite
Descriptive Title of your Review Paper: Automatic Parking Detection

In this fast-paced world no matter adolescence or adults, people only want to finish what they have on their hands and mind their own business. To cope with the fast-paced world, people have come up with many successful ideas. For example, Uber is a great app that will help someone to find a taxi immediately. Another example, Doordash and Uber Eats are also apps that can help with a fast meal if there is no time to cook. The list goes on and on because to save as much time as possible people choose simple and fast options that will give them the most optimized result.

Now imagine this, a car is pulling into a busy Disney World parking lot without proper direction, just think how long this car will spend searching for a spot. Another problem that can happen is parking spots hidden in between multiple cars, which is really difficult to detect. The most common parking system, if there is one, is labor-intensive, needing people standing in the lot giving directions on where to go. The proposed idea is a set of technology that will solve the mundaneness in parking.

The idea is very simple. Cameras will be set up in a convenient location that can have a clear view of the entire parking lot. Then the cameras find out the edges of the parking lot, the empty spots, the size of the spot, and if the spot is special (disabled spot for example). When a car is driven into the entrance a display screen will be there, where the fastest route to get to the nearest fitting parking spot will appear. A camera will be placed at the entrance to determine the car type, for example, SUV, pick-up, Sedan, Van, or even a truck/RV. If the driver has a tag indicating a staff or handicap status, the software can direct you to a special spot. After this system is set up the labor cost will go down dramatically. A picture of the map, if someone can not remember a longer route, can be sent via Bluetooth, or the driver or passenger can just take a picture. Similar ideas have been made and implemented by Parking Guidance System, LLC, and tn. However, all of these companies only work in a garage and they need heavy modifications to the parking garage. For example, Hub Parking requires an LCD screen in every intersection/turn inside the garage and sensors on every single spot. The cost of modifications to the garage weighs more than the benefits (Kenneth). Since most parking garages/lots already have security cameras set up there is very little to modify. The only required addition to the parking space is a screen at the entrance or a camera with a better angle of viewing. Plus to have AI working in a garage is very futuristic.

To get the Automatic Parking Detection working both hardware and software are required. The cameras will be connected to a central computer where all the data and CNN do the magic. The reason

why there is no microcontroller for this problem is that running/training models on a Mbed or a Pi cost too much power and time. Pi has an OS system for the convenience for users, however, adding an OS will increase the time of your program. Getting everything connected to a central computer with a powerful GPU and CPU will make everything easier. If multiple cameras are needed imagine how much it would cost just for these microcontrollers. If there is a system crash on these controllers a walk to the camera might be required, which is more trouble than it's worth (Halfacree). For the car detection parts, there are already applications/algorithms online, for example, a python library called "deep-learning-frameworks", which is available on GitHub. This library can help detect cars from a bird's eye view, and determine the size of the car (Clee). With Esri's Python library a lot of steps are saved, segmentation of the image, for example, is already here. YOLO is another object detector that can give an accurate description of the type of the car, for example, it can tell apart a truck and a Sedan (Redmon). To add more car types into the system is a challenge when implementing the idea since the camera should be able to recognize a pick-up truck from a Minivan, which is a little difficult. Another thing that is required is finding the edge of the empty spot for a parking lot. Fortunately, OpenCV, an image processing library in both C++ and python, can be used to help simplify the problem (Murkute). Of course, without Tensorflow deep learning is just not right. With the combination of Tensorflow and OpenCV, wonders can be achieved.

So currently, there are four different ways this can be done. The first option is to use just OpenCV to check for an empty parking spot. Using OpenCV is very simple since the library is already very defined and the functions are mathematically logical. The only problem with this method is that lighting will play a huge role, meaning different algorithms might be needed for day and night time. The second solution is to use YOLO's object detection to identify all cars and then check whether this car can fit inside a parking spot. The drawback of this option is having to add more car types and match the parking spot size with the car size. The prediction for this method is 70-80 accuracy. The third method is to use a CNN which comes from the Tensorflow library. This method will probably end up being the most accurate according to research. The fourth option is to combine all the libraries and make them work concurrently to get the most optimal result (Dwivedi). The design will utilize more than one functionality listed, so option four will most likely end up being used. YOLO is used for objection detection for size and type, OpenCV is used for empty lot detection, and CNN for prediction. To test our model, some real-life parking lot images will be used. The image will be either from Google or real-time capture done by the team. The score will be rated on a scale of accuracy, meaning 0-100% with a target range of 90% or more.

Fast, accurate, and convenient are the goals the team is trying to achieve. Making parking effortless for everyone.

A. Murkute, “Parking lot detection techniques,” *Medium*, 29-May-2020. [Online]. Available: https://medium.com/car-parking-assist-prorotype/parking-lot-detection-techniques-7792e84febbf. [Accessed: 07-Oct-2021].

B. Clee, “Car Detection - USA,” *Arcgis.com*, 09-Sep-2021. [Online]. Available: https://www.arcgis.com/home/item.html?id=cfc57b507f914d1593f5871bf0d52999. [Accessed: 07-Oct-2021].

G. Halfacree, “Raspberry pi 4 b: How much ram do you really need?,” *Tom's Hardware*, 02-Jun-2020. [Online]. Available: https://www.tomshardware.com/news/raspberry-pi-4-how-much-ram. [Accessed: 07-Oct-2021].

J. Redmon, “YOLO: Real-Time Object Detection,” *Yolo: Real-time object detection*, 2018. [Online]. Available: https://pjreddie.com/darknet/yolo/. [Accessed: 07-Oct-2021].

Kenneth and P. Articles, “Parking guidance systems,” *PARKING GUIDANCE SYSTEMS, LLC.*, 02-Mar-2020. [Online]. Available: https://parkingguidancesystems.com/. [Accessed: 07-Oct-2021].

P. Dwivedi, “Find where to park in real time using opencv and tensorflow,” *Medium*, 27-Mar-2019. [Online]. Available: https://towardsdatascience.com/find-where-to-park-in-real-time-using-opencv-and-tensorflow-4307a4c3da03. [Accessed: 07-Oct-2021].

Tensorflow , “Convolutional Neural Network (CNN) : Tensorflow Core,” *TensorFlow*, 2021. [Online]. Available: https://www.tensorflow.org/tutorials/images/cnn. [Accessed: 07-Oct-2021].