Convolutional neural networks (CNNs) are a type of layered deep neural network comprised of artificial neurons. These neurons are initially taught a set of rules and conditions, through training, which dictate whether they will fire when given varying inputs. CNNs learn as they are used and make future decisions based on both the taught and learned information. A common application of CNNs is object and feature recognition in images. The CNN identifies features in an image by analyzing data pixels through layers of neurons. This is particularly useful in the field of autonomous vehicles where CNNs can be used to process driving footage and identify possible obstacles. CNNs will often classify sections of the preset image grid that potentially contain an obstacle. Errors that occur are fed back into the network for reclassification and further learning. After the analysis is complete and a final conclusion has been reached, the CNN outputs a signal for the vehicle to perform an action: keep driving, stop, turn, etc. NVIDIA tested the use of CNNs in an autonomous car in 2016. While there have been recent advances in the field of autonomous driving, some issues have arisen with regards to the ethics of using a computer to determine the outcome of a collision. It has been argued that it is not yet safer to have a CNN controlling a vehicle than a human driver. On the other hand, autonomous vehicles promise to increase the sustainability of the common driving environment. With more research in areas such as the safety and ethicity of CNNs, this technology could fuel the next transportation revolution.

**Key Words**—Convolutional Neural Networks, Unmanned/Autonomous/Self-Driving Vehicles, Image Recognition, Obstacle Detection, Depth Estimation, Deep Learning, Machine Vision

Deep neural networks are computerized decision-making networks that mimic the mammalian visual cortex. The structures of deep neural networks consist of multiple layers of neuron-like components. Their layers allow them to be extremely versatile because they can process inputs through multiple parameters. Subtypes of these networks include convolutional neural networks (CNNs) and deep belief networks (DBNs). Convolutional neural networks are traditionally used for image analysis and object recognition. In the past decade, there has been an effort to expand the applications of CNNs, specifically, their use in autonomous vehicles for object detection and depth estimation. In 2016, NVIDIA created an autonomous car using CNN technology. Their car exemplifies and demonstrates the validity of using CNNs in autonomous transportation. Employing CNNs in vehicles has the potential to improve road safety, but this comes with questions about the ethics and liabilities of a computer making decisions in a crash situation. It has not yet been found that a computer driven vehicle could actually keep passengers safer than a human driver. However, the implementation of decision making in autonomous vehicles will create a more sustainable driving environment. In this case, sustainability means there will be decreased negative impacts on the environment and preservation of the vehicle and its parts so as to give them a longer lifespan. The decreased environmental impacts come from optimized fuel usage and minimized wear on vehicle parts allowing for a decrease in manufacturing. Overall, CNNs are still a new technology, but have promising applications in autonomous driving.

**STRUCTURING A CONVOLUTIONAL NEURAL NETWORK**

**Neuron Structures and Basic Functions**

A neural network is essentially a collection of artificial neurons arranged into layers with inputs from one layer being passed on to the next layer as outputs. One type of artificial neuron is a perceptron. Inputs are weighted based on their importance to the final outcome and sent to the perceptrons which fire according to a binary system. Each perceptron has a given threshold that decides if it will fire. If the sum of the weighted inputs is greater than the
The perceptron's threshold, the output is one and the perceptron fires. When the threshold is moved to the other side of the inequality, it is renamed the bias where \( \text{bias} \equiv -\text{threshold} \) [1]. This is represented by the equation in figure 1 where the sum of all the inputs and their respective weights is written as \( w \cdot x \) and \( b \) is the bias.

\[
\text{output} = \begin{cases} 
0 & \text{if } w \cdot x + b \leq 0 \\
1 & \text{if } w \cdot x + b > 0 
\end{cases}
\]

**FIGURE 1 [1]**
Equation of a perceptron’s firing potential

This equation shows that with a large positive bias, it is easier for the perceptron to fire and harder for it to fire with a large negative bias. As layers of perceptrons are added, the network is able to make more complex decisions. However, the outputs are binary, so changing the weight of one perceptron can greatly change the network's final output [1].

When modifying a neural network, it is beneficial to be able to make small changes to the weights and biases without drastically changing the final output data. These small changes can be created using sigmoid neurons. Sigmoid neurons have an almost identical structure to perceptrons, with multiple weighted inputs, a bias, and one output. However, the output of a sigmoid neuron can be anywhere between 0 and 1 which means that small changes in the input weights and neuron biases can create small changes in the network’s output. Figure 2 shows a graphical representation of the difference between the outputs of a perceptron and a sigmoid neuron.

**FIGURE 2 [1]**
Graphs of outputs from a perceptron (left) and a sigmoid neuron (right)

Using the gradated output of a sigmoid neuron is useful when evaluating something such as the intensity of pixels in an image, but not as useful when deciding whether the image is or is not of a specific object. To allow for simple true-false identification, the sigmoid neuron can be set to fire (true) if the output is greater than 0.5 and not to fire (false) if the output value is less than 0.5 [1]. Changes to the sigmoid neuron’s input weights and biases are made during the learning stage of the neural network.

While the network has an initial configuration, the learning algorithm changes the network as it is taught using a set of training data. For image recognition networks, the training data is composed of images. As the network is fed training data, it improves its parameters by using both stochastic gradient descent and backpropagation. Before these learning methods can be discussed, it is first necessary to explain the layered structure of CNNs.

**Layers of a Convolutional Neural Network**

Convolutional neural networks contain multiple types of layers through which all data is fed. These layers are arranged in a hierarchical manner and can include convolution layers, pooling layers, fully connected layers, and a loss layer. Each layer has its own focus and purpose in the process of analyzing data. With every successive layer, the analysis becomes more abstract. In image recognition, this means the first layers react to stimuli such as oriented fields or changes in light intensity, while later layers determine the identification of an object and make intelligent decisions about its importance. This is a large generalization of what the layers “look for” in an image. The layers actually process each pixel of the image based on the mathematical functions in their neurons. While all layers are made of neurons, not all layers serve the same purpose.

The convolution layers are the most involved layers. To understand how these layers work, it is important to first understand the concept of convolution. Convolution is officially defined as the integral that shows the amount of overlap of one function as it is moved across another function [2]. In our case, the convolution part of the network refers to a representative kernel being moved across a given image. A kernel is a matrix of values created to detect different features. For example, you may have a 3x3 kernel which is assigned nine separate values. These values are assigned so that the kernel has a specific function. This kernel is then successively centered on each pixel of the image. As the kernel is iterated across the image, the inner product of the kernel and the overlapping image is calculated and that value is assigned to the corresponding pixel. This value is a measure of similarity between the input image and the kernel. The application of a kernel to an image is demonstrated in figure 3 where the pixel being analyzed is referred to as the source pixel and the pixel in the output image is the destination pixel.
In autonomous vehicles, the kernels are designed to detect specific features such as edges. When the convolution reaches a local maximum, that position is classified as an edge [4]. Even within edge detection, different kernels are needed. Depending on their values, kernels can be used to detect differently oriented edges. Figure 4 shows an initial image alongside examples of different edge detection kernels and how they affect that image.

As shown in figure 5, each 4x4 subregion is condensed to one value and the outcome is a 4x4 matrix which represents the original data. In a CNN, this smaller matrix allows for faster processing as it is passed to future layers. In the applications discussed in this paper, the most commonly used pooling method is max pooling. By using max pooling, insignificant data points such as those not containing an edge (lower likelihood of containing an obstacle) are eliminated and the future layers can easily analyze regions with higher risk. A pooling layer is thus a way to focus the CNN on regions which contain potential obstacles.

Fully connected layers usually come at the end of the network to condense the data to one final output. Fully connected layers function similarly to convolution layers in that they both produce inner products. However, every neuron in a fully connected layer is connected to every output of the previous layer whereas in the convolution layers, neurons are connected to groups of outputs. This means the fully connected layer analyzes all of the data points simultaneously without performing a convolution function. Figure 6 shows a typical CNN structure starting with convolution layers of increasing intricacy and ending in a fully connected layer to condense the data.
By placing the fully connected layer at the end, the network is able to collect in-depth data in many dimensions through its convolution layers, but condense this information into a readable output in the final fully connected layer.

The loss layer comes at the end of the network and functions as an error feedback loop. This type of layer is only needed for problems involving learned parameters such as when using training examples [4]. The loss layer is used mainly in backpropagation and is where the cost calculations occur once the network has run through a set of inputs. One important note about the loss layer is that it is not a permanent aspect of the network. After the CNN has been fully trained and is ready for experimental or practical use, the loss layer is no longer needed because the network is assumed to be at its maximum accuracy at that point.

Learning in a Convolutional Neural Network

Convolutional neural networks learn using stochastic gradient descent and backpropagation. The goal is to make the predictions of the CNN match the ground-truth (original input image) by minimizing a cost function. To train a CNN, it must be run in both a feedforward and feedback configuration.

Over the forward run, small errors are collected and analyzed by the loss layer. The cost shows the margin between what the CNN recognizes and the ground-truth. This error is minimized using stochastic gradient descent. Stochastic simply means that the training images are fed through the network in small, random subsets [4]. A subset method is better than feeding a whole set at once because it allows for variety in what the network sees. If the images were not fed in subsets, the network would only be able to identify images in the specific configuration of the training set. Gradient descent is the more complex part of this process.

Gradient descent is a process used to minimize error similar to the least squares method. To apply gradient descent, there must be a cost or loss function involved. This is the function that is being minimized. The equation in figure 7 is an example of a cost function taken from Michael Nielsen’s book, “Neural Networks and Deep Learning.”

\[ C(w, b) = \frac{1}{2n} \sum_x \| y(x) - a \|^2. \]

Nielsen defines the variables as “\( w \) denotes the collection of all weights in the network, \( b \) all the biases, \( n \) is the total number of training inputs, \( a \) is the vector of outputs from the network when \( x \) is input, and the sum is over all training inputs” [1]. In this equation, \( y(x) \) represents the desired output and \( a \) is the output the network has provided. As you can see, the closer \( a \) gets to \( y(x) \), the smaller the cost \( C \). Gradient descent is the algorithm used to find the collection of weights and biases that will allow \( C \) to reach its minimum value. The main idea behind the gradient descent algorithm is to find the gradient of the function to be minimized and move the values of the variables in the opposite direction so as to guarantee the gradient will always be decreasing. This is the point where backpropagation comes in.

Backpropagation occurs when the network is run in the backward direction to adjust the parameters of each layer. In the case of a CNN these parameters are the kernels. The loss layer collects the errors and analyzes them with the gradient descent algorithm. Then, adjustments for the parameters are propagated back through the layers of the network. Figure 8 illustrates errors being sent back through a sample network.
IMAGE RECOGNITION

Convolutional neural networks are applied in many image recognition problems. However, they are currently used mainly in the identification of specific objects. This means that, when used in a car, the network analyzes the input image and finds areas that have a specific feature such as another car or a pedestrian. In order to classify an object within an image, the CNN must know what to look for. This is why the network is given a set of training images. The general hierarchy for the identification of an image is as follows: pixel → edge → texton → motif → part → object [8]. Pixels and edges are just as one might expect. Textons are “fundamental micro-structures in generic natural images and the basic elements in early (pre-attentive) visual perception” [9]. They are small sections of texture or pattern which are then combined into motifs. Motifs are sections of repeating patterns that can later be combined into larger image parts [10]. These parts are then combined to form a whole image to be identified. Because the network has been trained, it can use its set of specific kernels to identify this image.

The first step in classifying an image is to break it up into sections, typically pixels, however these sections can be larger than pixels. The image is then analyzed by the CNN. Each layer’s kernels are designed to extract specific features. These kernels follow the previously mentioned hierarchy and become more intricate and specific as the layers progress. An example of a set of convolution kernels increasing in complexity is shown in figure 9a.

![Kernels become more intricate with each layer](image)

FIGURE 9a [11]

Positioning of kernels within a facial recognition CNN

Each grid square in figure 9a represents one kernel which is passed over each pixel of the image. The final output is a representation of the original ground-truth image. Figure 9b depicts a set of kernels and their respective positions in a sample facial recognition CNN architecture [8]. As the layers progress, the kernels grow closer to representing the structure of a human face.

Currently, most CNNs are designed to classify a specific object such as handwriting, faces, or animals. For CNN classification to be effective in autonomous vehicles, the network needs to be able to classify a diverse set of objects or be able to recognize simply what is a potential obstacle and what is not.

CONVOLUTIONAL NEURAL NETWORKS IN AUTONOMOUS VEHICLES

Obstacle Detection

Image identification in autonomous vehicles is not as simple as facial or handwriting recognition because vehicles need to process a full 360 degree dynamic environment. This creates the need for dual frame processing because collected frames must be combined and considered in context with each other. A vehicle can be equipped with a rotating camera to collect all relevant driving data. The machine must be able to recognize metric, symbolic, and conceptual knowledge as demonstrated in figure 10.

![Metric Knowledge (yellow), Symbolic Knowledge (orange), and Conceptual Knowledge (red) applied to a driving scene](image)

FIGURE 10 [12]
Metric knowledge is the identification of the geometry of static and dynamic objects. This is required to keep the vehicle in its lane and a safe distance from other vehicles. Symbolic knowledge allows the vehicle to classify lanes and conform to basic rules of the road. Conceptual knowledge allows the vehicle to understand relationships between traffic participants and anticipate the evolution of the driving scene [12]. Conceptual knowledge is the most important aspect for being able to detect specific objects and avoid collisions.

One current method of obstacle detection in autonomous vehicles is the use of detectors and sets of appearance-based parameters. The first step in this method is the selection of areas of interest. This process narrows down areas of the field of vision that contain potential obstacles. Appearance cues are used by the detectors to find areas of interest. These appearance cues analyze two dimensional data and may be sensitive to symmetry, shadows, or local texture and color gradients [12]. Three-dimensional analysis of scene geometry provides greater classification of areas of interest. These additional cues include disparity, optical flow, and clustering techniques. Disparity is the pixel difference for an object from frame to frame. If you look at an object and alternate closing one eye after the other, the “jumping” you see in the object is the disparity. It can be used to detect and reconstruct arbitrarily shaped objects in the field. Optical flow combines scene geometry and motion. It samples the environment and analyzes the images to determine the motion of objects. Finally, clustering techniques group image regions with similar motion vectors as these areas are likely to contain the same object [12]. A combination of these cues is used to locate all areas of interest. While any combination of cues is attainable, it is necessary to include both appearance cues and three-dimensional cues as the accuracy of three-dimensional cues decreases quadratically with increasing distance [12]. In addition, only persistent detections are flagged as obstacles so as to lower the rate of false alarms. After areas of interest have been identified, they must be classified by being passed through many filters that search for characteristic features of on-road objects. This method takes a large amount of computation and time. The use of CNNs can increase the efficiency of this detection process.

Baozhi Jia, Weiguo Feng, and Ming Zhu, researchers at the University of Science and Technology of China, developed a CNN-based detection system that can classify areas that contain any type of obstacle. They argue that motion-based methods such as optical flow are too heavily reliant on the identification of feature points, which are often misclassified or not present in the image. Concerning current neural network and other knowledge based approaches, they say, “All of these knowledge-based methods are for special obstacles (e.g., pedestrians, cars) or in special environments (e.g., flat road, obstacles differ in appearance from ground)” [13]. Convolutional neural networks are the most promising method for classifying complex scenes because they closely mimic the structure and classification abilities of the human brain. The researcher’s model includes a CNN for classification of local features (recognizing known obstacle types such as pedestrians and trees), as well as a deep belief network (DBN) for global feature interpretation (shape contours and textures). The DBN functions similarly to a CNN, but is taught to interpret large-scale global features for object detection in an image, whereas the CNN is used for specific object identification and decision making.

Using a combination of a CNN and DBN, the researchers were able to classify areas containing any type of potential obstacle. The entire network was trained using the Daimler Urban Scene Segmentation Benchmark (DUSSB) and Karlsruhe Institute of Technology and Toyota Technological Institute (KITTI) databases of on-road images. It was then tested using campus road images (CRI) collected by the researchers through a camera on a campus navigation car [13]. Figure 11 shows their test results.

**FIGURE 11** [13]
Obstacle detection test results: input images (top), ground truths with black as positive (middle), and detected obstacles with orange as positive (bottom)

Additionally, when compared to other obstacle detection methods, the CNN and DBN combination method proved to be more accurate as shown in figure 12.

**FIGURE 12** [13]
Results from CNN/DBN network (our approach) compared to other common methods with positive detections in orange

Obstacle detection is only one important part of avoiding a collision. It is also vital for the vehicle to recognize how
far away the obstacles are located in relation to its own physical boundaries.

**Depth Estimation**

Estimating the distance between an obstacle and the vehicle is an important safety concern. A CNN may be used for this task as well. In a study done at the Technical University of Berlin, Ahmed J. Afifi and Olaf Hellwich found that CNNs are a viable method to estimate depth in an image. They trained their network on A Large Dataset of Object Scans, which is a public database of over ten thousand scans of everyday 3D objects [14]. They focused on images of chairs and used two different loss functions for training: Tukey’s biweight loss and L2 norm [15]. The networks were then tested on the Ikea chairs dataset and the results are shown in figure 13.

![Figure 13](image)

**FIGURE 13 [15]**
Results of a test measuring depth in pictures of Ikea chairs: blue is close, red is farther away

The Tukey’s biweight trained network was more accurate at finding depth than the L2 norm. With images of varying size and resolution, it had an accuracy of between 0.8283 and 0.9720 with a perfect accuracy being 1.0 [15]. Afifi and Hellwich thus showed that a CNN is an effective method for depth estimation.

While estimating depth on single frame, stationary objects is simpler than on the moving objects seen by vehicles, Jia, Feng, and Zhu’s previously mentioned study proved CNNs can also be used for depth estimation in driving scenes. Their method consisted of feeding detected obstacle blocks to a second CNN programmed to find depth. The blocks were split into strips parallel to the lower image boundary. These strips were weighted with depth codes from bottom to top with the notion that closer objects would normally appear closer to the lower bound of the image. The depth codes went from one to six with one representing the most shallow areas and six representing the deepest areas [13]. The obstacle blocks were assigned the depth code for the strip they appeared in. The CNN then used feature extraction in each block area to determine if vertically adjacent blocks belonged to the same obstacle. If the blocks were determined to be the same obstacle, they were assigned the lower depth code to alert the vehicle of the closest part of the obstacle [13]. In this study, the CNN was trained on image block pairs to develop a base for detecting depth and then tested on street images as it was for the obstacle detection method. The CNN had “an accuracy of 91.46% in two-block identification” [13]. Figure 14 shows a sample of test images and the depth assignments as decided by the CNN.

![Figure 14](image)

**FIGURE 14 [13]**
Results of a test determining depth in images with a CNN

Depth estimation is an important consideration in autonomous driving as it ensures the safety of the passengers and of other vehicles. It is aspects of CNN usage like this that have been applied in projects such as NVIDIA’s autonomous car.

**NVIDIA’S AUTONOMOUS CAR**

In 2016, NVIDIA developed an autonomous car using CNN technology. The project involved collecting training data, allowing the car’s CNN to learn from the training data, using a simulation to test the car, and finally driving it on actual roads. The network was trained on dozens of hours of collected driving footage. The training data included footage from three front-facing cameras as well as related steering commands. As the data was fed into the CNN, it learned to output the appropriate steering command for different situations. This process is shown in figure 15.
Once the CNN had been trained, it was equipped with one front-facing camera as opposed to the three that were used to collect driving footage. The CNN could then interpret the features of a single stream of images to determine whether it was maintaining an obstacle-free path on the road. However, before the autonomous car was allowed to drive on real roads, it was tested in a simulation. The simulation used previously captured road footage as initial input data and then used the CNN’s outputs to produce a synthesized image of what the vehicle would see based on the decisions it made. The simulation also included a human intervention feature that activated whenever the car veered more than a meter off its desired path. Thus, the simulation provided a realistic preview of how the autonomous car would perform on actual roads. Figure 16 shows an outline of the simulation program.

After sufficient simulation testing, the car was driven on physical roads. Again, a human driver sat in the driver’s seat to intervene if the CNN failed to keep the car on its path. The CNN used by NVIDIA was never explicitly taught to recognize lane lines or road edges. It learned through the training process what to look for in the input images as significant and relevant information for navigation. This means the network can draw on previously acquired data to navigate new situations that don’t necessarily have explicitly defined features. Figure 17 shows how the CNN analyzes a situation with distinct path features versus a situation with very few navigation indicators.

NVIDIA’s project produced a CNN-driven car that was fully autonomous during 98% of the on-road testing, meaning human intervention was only required 2% of the time [16]. While this is a promising result, there are still ethical questions to be answered before autonomous vehicles can be put to widespread use.

### WIDESPREAD USE OF AUTONOMOUS VEHICLES

#### Ethical Decision Making

A common misconception is that autonomous vehicles will provide a safer, crash-free future for transportation. While this is a main goal of autonomous transportation, statistics have yet to support this claim. Although computerized systems can compensate for human errors such as emotional distractions and insufficient reaction times, collisions cannot yet be completely avoided. There are some factors that a computer cannot predict in a crash...
situation. Examples of these factors include natural disasters such as earthquakes or landslides as well as human made disasters like bridge collapses. In a potential crash situation, the CNN is responsible for making two decisions: whether a collision is going to occur on the vehicle’s current path, and if so, which new path to take.

The decision-making process for autonomous vehicles is complex and can sometimes fail to prevent the crash in an unexpected or unpredictable situation. Autonomous vehicles cannot yet consider numerous ethical factors including the safety of the passengers, the safety of the people outside of the vehicle, and human intervention. The network simply considers the scene features and what driving command to execute. Many nuances of these ethical factors are pushed aside in favor of assurances that the human in the driver’s seat will intervene and the car will not be required to take any action other than to alert the driver. However, in reality, it cannot be assumed that the driver will have the time and focus to react, or that they will make a decision that is better than that of the CNN.

Even if an autonomous vehicle system could be programmed in a way that would determine which path the vehicle should take in any given scenario, extensive testing would be necessary to provide evidence that autonomous vehicles are, in fact, safer than human drivers. According to mileage data collected in 2009, a CNN controlled vehicle would have to drive 725,000 miles without incident or human intervention to maintain a 99% confidence that it can drive more safely than a human driver [17]. While there have been many advances in autonomous vehicle technology, more testing is needed before it will create a legitimately safer driving environment.

Development of a Sustainable Driving Environment

While a crash-free driving environment has not yet been achieved, there is potential for such an occurrence in the future. However, the avoidance of crashes is not the only advantage to making autonomous vehicles a common form of transportation. These vehicles will provide sustainability to the environment and to the vehicles themselves.

Autonomous vehicles provide sustainability because they have the ability to decrease energy consumption and wear on vehicle parts. A mechanism for deciding on an ideal route can be included in the neural network of each vehicle. This route optimization and resulting decrease in traffic congestion is predicted to reduce fuel consumption by up to 4% therefore reducing the amount of ozone and environment harming emissions [18]. Additionally, as in NVIDIA’s autonomous car, each vehicle will have learned a set of commands for different driving situations. When efficiency and part preservation are prioritized, the commands will be executed in such a way as to minimize wear on the vehicle and reduce energy consumption by 25% [18]. For example, if a human driver is stuck in traffic, they might hit the accelerator and then the brake excessively to move every time the traffic inches forward. This causes excessive wear on the engine and brakes of the vehicle. However, in an autonomous vehicle, the system would be optimized so that it either rolls forward at a slow enough rate that it will not collide with the vehicle in front of it, or it will not move until there is enough free road to justify doing so. This will decrease the wear on the vehicle’s brakes and engine as well as further optimizing fuel efficiency. As a result, the lifespan of each vehicle will be prolonged, thus decreasing demand for new vehicles and vehicle parts. The manufacture of less vehicles means the conservation of resources such as fuels burned in factories and metals used in production. Overall, the realization of an autonomous future for driving will help preserve natural resources and reduce emissions that are detrimental to the environment as well as prolonging the life spans of individual vehicles.

CONVOLUTIONAL NEURAL NETWORKS AS THE FUTURE OF AUTONOMOUS TRANSPORTATION

The trainable, multi-layered structure of CNNs is what sets them apart from other neural networks. Their layers can be customized by adjusting their parameters to best fit their intended function. The neurons are constantly improving the accuracy of their outputs by learning from each piece of input data. This is particularly useful for applications in autonomous vehicles. The capabilities of CNNs to distinguish both the presence and depth of obstacles makes them promising backbones for autonomous transportation. In 2016, NVIDIA was able to demonstrate these qualities with their autonomous car. However, the ethics of collision decision making still provide a considerable obstacle for the use of autonomous vehicles in everyday life as they have not yet proven to be safer than human drivers. On the other hand, these vehicles will promote sustainability by decreasing environmentally harmful emissions and reducing wear on vehicle parts. The more research that is compiled relating to CNNs in autonomous vehicles, the closer we are to introducing these vehicles as a main form of transportation.

SOURCES

Erin Welling
Maddie Oppelt

https://community.arm.com/graphics/b/blog/posts/when-parallelism-gets-tricky-accelerating-floyd-steinberg-on-the-mali-gpu


ADDITIONAL SOURCES


ACKNOWLEDGMENTS

We’d like to thank Oliver Snyder for explaining the foundations of neural networks and introducing us to useful articles regarding image interpretation. We would also like to thank our co-chair Ryan Becker and our chair Frank Kremm for their guidance and suggestions.