ASEE AMERICAN SOCIETY FOR ENGINEERING EDUCATION

Paper ID #35069

Integrated Project Platform for Student Research and Curriculum Development

Mr. Niklas Cyril Bitters
Dr. Ramakrishnan Sundaram, Gannon University

Dr. Sundaram is a Professor in the Electrical and Computer Engineering Department at Gannon University. His areas of research include computational architectures for signal and image processing as well as novel methods to improve/enhance engineering education pedagogy.

Work-in-Progress: Integrated Project Platform for Student Research and Curriculum Development

Abstract
This paper describes the setup of an integrated project platform to support student research and curriculum development in the burgeoning topics of artificial intelligence and machine learning. The platform comprises modules for object classification and collision avoidance which are used in the design of intelligent and autonomous vehicles. Object classification consists of the appropriate neural network architecture for training and learning object characteristics from data sets. Collision avoidance utilizes a single camera to estimate the distance of the vehicle from the object. The hardware and software requirements of the integrated project platform are met by cost-effective resources. In addition to serving as a testbed for student research in the design and operation of intelligent and autonomous vehicles, project activities on this platform will enable students to gain valuable laboratory and project experiences. This can be accomplished through the inclusion of exercises on this platform in graduate and undergraduate courses offered as part of the electrical and computer engineering (ECE) curriculum. Graduate ECE courses, such as image processing, neural networks, and embedded system design would be choices for project activities on this platform. Typical courses in the undergraduate ECE program are digital logic design and programming in C/C++/Python. The platform will promote student participation across the ECE program in competitive design events for the next generation of intelligent and autonomous vehicles. Evidence of the use of the platform and the assessment of learning outcomes will be documented in future papers.

Introduction

The engineering programs at universities across the world must adapt to the rapidly changing engineering technology and the needs of the global workforce. The engineering students who enroll at these universities expect to be educated and trained with the latest industry-approved tools in order to function effectively in the engineering industry. In recent years, artificial intelligence (AI), machine learning (ML), and the internet-of-things (IoT) have been able to reduce or eliminate human interaction while yet processing large amounts of data. AI offers computational tools that replace the need for humans to perform certain repetitive tasks. The industries which already use AI include health care, retail, manufacturing, and banking.

The adoption of an integrated project platform, henceforth labeled IPP, by the Electrical and Computer Engineering (ECE) program at Gannon University, Erie, PA is viewed as an effective approach to strengthen and broaden the education of the ECE student [1]. Specifically, the appropriate choice of the platform can accomplish the following.

Broad goals

- Train the student to think and work like an engineer
  - *emphasize concept to design across the curriculum*
  - *create the environment for goal-oriented and self-directed learning*
- Shape the student into a 'successful engineering entrepreneur'
  - *equip the student with the skills to function in a global business with economic uncertainties*

Specific objectives

- Use cost-effective resources to create the IPP
- Unify the content across the set of courses which advance the preparation of the student toward skill sets in AI, ML, and IoT
- Deliver laboratory exercises and team projects on the IPP which emphasize the problem-based and project-enhanced pedagogy in each course

The motivation behind the choice of IPP is as follows:

The Intelligent Ground Vehicle (IGV) Competition (IGVC) [2] requires the competitors to design and assemble a fully autonomous and unmanned ground vehicle capable of navigating a prescribed course with obstacles. The vehicle must also perform specific tasks assigned during the competition. The team of students in the ECE department have competed in the IGVC [3] in previous years. However, due to the limited time available to prepare for the competition, the vehicle presented by the team could not successfully complete all the tasks assigned in the competition. The IPP was conceived by one of the authors to resolve design issues and to create the environment for student teams to engage in the test and validation of proposed design changes. In this paper, the IPP setup comprises modules to implement the neural network, object classification, and collision avoidance.

Section 1 overviews the setup of the IPP with the top-level system description of the modules. Section 2 provides details of the operations in each module and the overall current status of the IPP. Section 3 discusses the application of the outcomes from the IPP to the IGV. Section 4 outlines the engineering course and curriculum development, engineering research activities, as well as co-curricular activities supported by the IPP. Section 5 comprises conclusions. Section 6 lists recommendations for future work.

Section 1: IPP set up

Figure 1 illustrates the modules for the top-level block diagram of the IPP. The first module is the Neural Network (NN) and comprises the steps taken to process the images of the obstacle. The images could be from a database of pictures of the obstacle identified as the training dataset or real-time streaming video (Webcam feed) of the obstacle. The NN used in our setup is the deep convolution neural network (DCNN) [4]-[6]. The DCNN performs a series of nonlinear transformations on the input image used for object detection or recognition. The final transformation outputs a vector of category probability values, one for each object category.

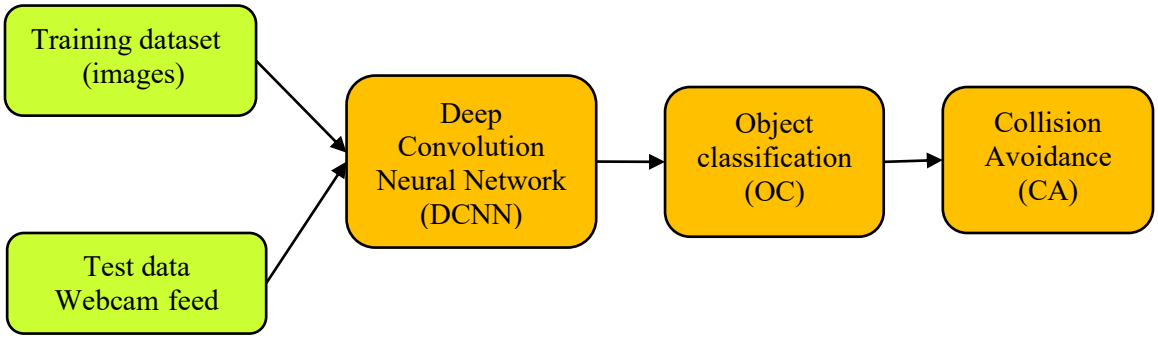


Figure 1: Top-level block diagram of the IPP

The early layers of these networks are not fully connected as in classical neural networks. Instead, convolutional windows are used to preserve the spatial information in the image [7]. In recent designs of the DCNN, the early layers tend to operate on very local regions of the image, while deeper into the network, each node receives input from filters over a larger area of the image. This allows the network to access relations between more distant regions [8]. This network architecture has some obvious similarities with biological vision.

The second module is labeled *Object Classification* (OC) and comprises the steps of detecting or locating and then labeling the object in each image. The objects are labeled by rectangular bounding boxes to show the confidences of existence. The IPP requires objects be identified in categories and locations. The regression or classification-based methods of object detection such as MultiBox [9], YOLO [10], YOLOv2 [12]-[14], Yolov3 [15]-[19], AttentionNet [20] are better suited than the region proposal-based methods such as R-CNN [21], Faster R-CNN [22], Mask R-CNN [23].

The third module labeled *Collision Avoidance* (CA) estimates the distance of the obstacle from the moving vehicle. Typically, the distance is calculated using a combination of proximity sensors or multiple camera feeds. However, in this paper the distance is accurately estimated using a single camera. The algorithm is implemented in Python to provide real-time measurements of the distance.

Section 2: Current status of the IPP

The traffic barrel has been identified as the obstacle to be classified and avoided. First, the IPP setup was trained to images of the traffic barrel as shown in Figure 2.

Training data set

Figure 2(a) depicts typical images of the use of the traffic barrel on roadways and highways. Figure 2(b) illustrates the use of the traffic barrels as obstacles along the course of the IGVC.

(a) (b)

Figure 2: Traffic barrel (a) highway uses (b) obstacle course

The IGV, which requires video in real-time, captures the feed using a single camera. The IGV can operate using visual acuity and data information as seen in Figure 3.

Test data – Webcam feed

Figure 3(a) exhibits the single camera utilized for obtaining the real-time webcam feed. Figure 3(b) features the complete IGV testing setup, which contains the secured webcam.

Figure 3: (a) Camera (b) Camera mounted on the IGV

Neural Network

The DCNN in the IPP is organized in five layers which are described as follows:

Convolutional layers

Seventy percent of the DCNN layers perform the convolution filter or kernel operation on the image of the obstacle. The filter or kernel operation is linear and involves multiplications of the image intensities with a set of filter or kernel coefficients (also known as filter weights). The filter, whose size (row and column dimension) is smaller than the size of the input image (e.g. 8x8 filter for a 256x256 input image), is applied repeatedly across the entire image. Different filters or kernels are used to create multiple feature maps [24]. Convolution layers are formed by applying filters not just to the raw image pixel values but also to the output of other layers. Stacking the convolution layers yield a hierarchical decomposition of the input image.

Short-cut layers

The short-cut layers (21.5% of the number of layers) represent layers that are bypassed or skipped so as to overcome the problem of gradient divergence and difficulty in training DCNN [25].

Route layers

The route layers (4% of the DCNN layer count) bring finer grained features in from the previous stages of the network. The route layers recover output from previous layers in the network and bring them forward to form the feature maps.

YOLO layers

In the YOLO layers (3% of the DCNN layer count), the coordinates of the bounding boxes, the class label and class confidence values are determined using the feature maps from the preceding convolutional layers.

Upsample layers

The Upsample layers (1.5% of the DCNN layer count) perform interpolation to recover the original size of the image.

Object Classification

The object classification module must recognize predetermined objects and generate bounding boxes around those objects. The bounding boxes must fit neatly around the specified objects, and not falsely classify objects. Also, the object classification must separately classify multiples of

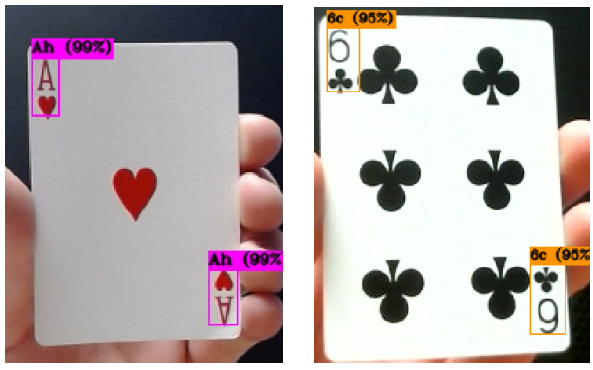
objects, not just a singular instance. This information will be obtained by the single camera [Figure 3(a)] and sent to the processing computer.

The Darknet framework [24] is adopted for object classification. Darknet is an open source neural network framework that was developed by the same individuals who pioneered the YOLO algorithm. For that reason alone, Darknet was the premier neural network choice for use in this project. In addition, Darknet does incorporate several additional advantages. Darknet is written completely in a combination of C and CUDA.

Training and testing the DCNN and Object Classification modules
The *DCNN* and *Object Classification modules* of the IPP are first tested with examples which consisted of identifying common everyday objects such as humans and cars. At this point, the network was able to receive input images or video, utilize the default weight classes, and establish bounding boxes around the humans and cars.

Once the examples were successfully implemented, the next step was to create a custom dataset and have the network identify custom objects. For this undertaking, it was decided to use playing cards as identifiable objects. The goal was to specify three unique playing cards within the network, which equates to three unique classes representing the three cards. This required the author to obtain three cards, and then take a few dozen pictures of each individual card. In order to provide the widest range of detection familiarity, the pictures had to portray different angles and degrees of lighting. In theory, the more pictures correctly classified, the better the accuracy of the neural network. Yet, the intention of this experiment was centered on the creation and detection from a custom dataset, using a reduced number of samples of each image.

Upon completion of the training of the neural network, the output custom weights can be used for testing. In order to do this, the configuration file must be slightly altered in a manner that initiates testing instead of training. Testing is essentially "running" the neural network with the weights found from training. In this scenario, the author could verify validity of the trained weights by testing the network and holding up individual playing cards to the camera. Therefore, the first attempt at custom dataset creation and implementation was a success. Figure 4(a) shows the correct identification of "Ah" for the ace of hearts. Figure 4(b) correctly displays the "6c" for the six of clubs.



(a) (b)

Figure 4: Playing cards identified (a) Ace of Hearts (b) Six of Clubs

Project dataset for the IPP

As mentioned previously, the primary motivation was to create the IPP as the testbed which is intended to be a successful marriage between deep learning methods and the IGV design for unmanned navigation. With this in mind, several choices and decisions have been made with both sides of the project in mind. One of those decisions was the fact to utilize standard traffic barrels as the primary dataset for training and testing the neural network. This required the author to obtain as many images of traffic barrels as possible. Therefore, the first method that the author performed was to collect traffic barrel images from the internet. Unfortunately, however, traffic barrel images are not as plentiful as traffic cones. In fact, there were only a few dozen clear traffic barrel pictures that could be found on internet browsers.

Due to the low number of traffic barrel images available on the internet, the author knew that personal pictures had to be included in the dataset. With that being said, one of the first tasks the author completed was obtaining physical traffic barrels. The Department of Transportation was contacted and five traffic barrels for research and educational purposes were granted. The author could use these barrels initially for dataset augmentation, and eventually, testing the network. An image of one of the traffic barrels collected can be seen below in Figure 5.

Figure 5: Standard Traffic Barrel

Labeling the traffic barrel

There are several prerequisite ingredients prior to actually labeling the dataset. First, the author sorted through all the images to make sure that they were of usable quality. Images that are too small or distorted will not offer much benefit. While encountering such low-quality images may not be an issue when capturing photos firsthand, it can be relatively common when pulling images from the internet. Second, the author removed any duplicates that were within the dataset. Duplicate images offer little, if any, benefit to the learning aspect of neural networks. Also, a large number of duplicate images can skew performance metrics. Lastly, it was essential for the author to scan the images for any irrelevant images. Once again, any personal pictures utilized would be relevant, but outside images may not be. After the images were scoured for quality control, it was then time to label the images.

While some scripts may manage automating a portion of the labeling process, the author did not search or use any scripts. Therefore, the author completed all labeling manually. Doing so involves looking at each individual image and setting a type of boundary box around the object in question. In the case of this project, the boundary boxes were placed around the traffic barrels

in each image. As this is done, there are associated text files that document the coordinates of the chosen boundary boxes. This tells the machine that the area within those coordinates contains an item of interest.

This project is only concerned with traffic barrels, which means that only one class has been outfitted in the neural network. If the project were concerned with detecting other objects, the classes added into the network would correlate with the object types. Still, it is important to note that this is not the same as object number per each image. If a single image contains four different traffic barrels, the author had to align individual label boxes around each of the four traffic barrels. With that being said, often images had traffic barrels that were partially visible or overlapping other barrels. It was still important to fit the label boxes around the parts of the traffic barrels that were visible. Likewise, it is important to set the label boxes as close to the image as possible. A labeled image together with the annotation of a traffic barrel is shown in Figure 6. The labeling tool used is called "LabelImg."

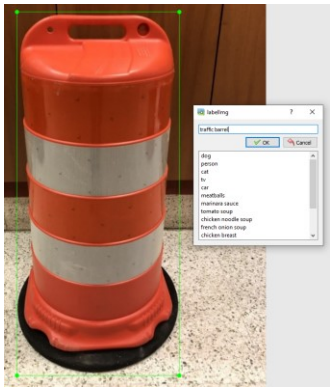


Figure 6: Traffic barrel – labeled and annotated

Classification results

Upon labeling all of the traffic barrels, training the dataset, and configuring all of the proper files, the final version of the object classification system was created. Anaconda script was used to navigate to the proper directory and launch the testing of the neural network. As soon as the Anaconda script is running and the camera has been turned on, it is possible to verify that the neural network is classifying objects correctly. It is important to note that the network is only going to classify images that were input into the system. Figure 7 shows two traffic barrels being correctly classified at the same time.

Figure 7: Classification of two traffic barrels

As mentioned previously, when labeling, it is also important to label objects that may be somewhat hidden from view. Since objects will not always be fully visible in real life, this trains the network for those instances. Figure 8 shows three traffic barrels being correctly classified, although there is overlap between two barrels.

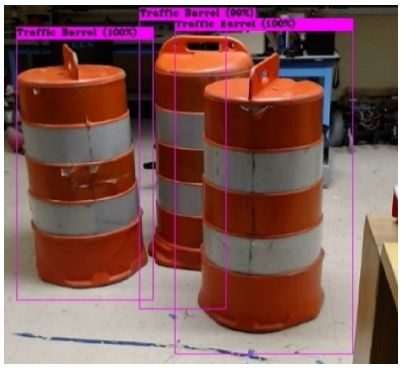


Figure 8: Classification of overlapping traffic barrels

Collision Avoidance

In the context of the IGV, the detection and classification of the obstacle is made more useful if followed by collision avoidance. Collision avoidance is performed with a series of spatial and distance awareness calculations made on the output from the object classification module. The distinguishing feature of this project is that collision avoidance is achieved using data from a single camera and no external sensors. Only the visual camera information has been used for the distance algorithm.

Single-Camera Distance Algorithm

The measurements would have to be done through pixel counts. The height and width could be measured for an object at a known distance, and then calculated for other distances. This concept is essentially triangle similarity, only the author would use variables such as pixels of an object and the focal length of the camera as described in the following equation.

$$F = ((P \times D))/H \tag{1}$$

where,

F = “Focal length, in pixels”
P = “Total pixel number of object width”
D = “Actual distance of the object, in centimeters”
H = “Real height of the object, in centimeters”

Since the author is using standardized traffic barrels, the height, H of all of the objects will be the same. The total pixel number of the object, P was determined by taking a picture of a traffic barrel from a known distance. The pixel width was then found by opening the picture and finding the range of pixels for the width of the traffic barrel. Meanwhile, since the picture of the traffic barrel was captured from a known distance, all of the variables are accounted for. This allowed one to solve for the focal length, F.

The calculated focal length is static and only applies for that particular distance of the traffic barrel from the camera. However, this is when the algorithm can be utilized in a converse manner. Now that the focal length, F had been established for a known distance, D, the equation (1) is used to estimate the distance, D using the calculated value of the focal length, F.

Validation of the Algorithm

In order to ensure that the object classification and obstacle avoidance components would connect in terms of code, it was decided to essentially recreate the project in Python [25], without a neural network. In light of this, developing a Python version of the project served two separate objectives. First, it allowed one to test the algorithm and determine if the pixels served as a feasible approach to calculating distance. Second, it allowed for continued productivity while rationalizing how the neural network version would culminate in the object classification and obstacle avoidance phases. Noting that Python is a popular programming language, there are an abundance of packages and documentation. This proved to be very convenient to create a Python script for image processing. With this in mind, OpenCV [26] proved to be exactly what worked. OpenCV is a library of mainly computer vision and image processing functions. Upon installing OpenCV to the machine, it was possible to incorporate the power of OpenCV image processing into a Python script through the command "import cv2."

In intentionally avoiding a non-neural network approach for the preliminary Python program, a different method of classification would be required. The simplest method proved to be color recognition. The Hue/Saturation/Value (HSV) representation for color discernment was used. Additionally, setting limits constrains the color recognition to occur only for the orange of the traffic barrels. It is also important to note that the Python program will detect the orange color that is in the greatest single area. In other words, if there are multiple orange features in the camera view, the largest one will be the one detected. Figure 9 demonstrates how the largest continuous orange area of the traffic barrel is bounded.

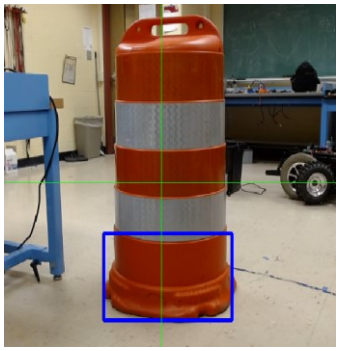


Figure 9: Color detection by the Python program

The Python program detects the largest orange area within the camera feed. The entire cone is not detected or accounted for, as seen in Figure 10. The camera input is continuously refreshed to provide estimates of the pixel width, P. the only variable that would change in the case of solving for distance, D. Figure 10 shows the traffic barrel at a distance of 143 centimeters from the camera. When measured alongside a tape measure, the actual result was only a few centimeters off. The blue boundary box and the yellow text with the real-time distance value is all part of the OpenCV library. The green lines are placed to facilitate the identification of pixel quadrants.

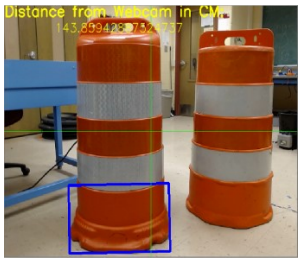


Figure 10: Detection and Distance Estimation by the Python program

Object Classification and Collision Avoidance

The success of the Python-based test program for color detection and obstacle avoidance enables one to consider a method to incorporate the same concepts in the actual neural network version. The goal is to determine how to logistically combine the object classification and object avoidance in one program. The approach to use purely Darknet C files was unsuccessful because of the difficulty to identify and edit the file for the placement of the bounding box. The DCNN weights and configuration files are accessed by the Python-based execution program. The boundary boxes are then created for the entire traffic barrel because these would be based on the training of the neural network and not on the color of the largest subregion of the image (see the blue box in Figure 10). Therefore, one would have to apply the distance algorithm for an entire barrel, and then once again obtain the pixel data. The system which includes the object classification and obstacle avoidance has been successfully developed. Figure 11 shows the traffic barrel with a bounding box and the distance value, measured in centimeters, from the web camera. The real-time rate of the program is about 12 frames per second (FPS). While this is slow for high-speed detection, most projects would find this detection rate serviceable.

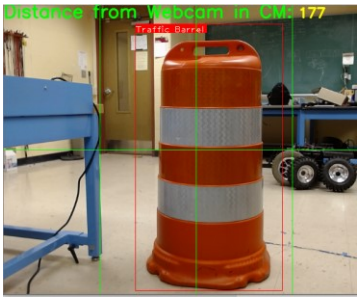


Figure 11: Object Classification and Obstacle Avoidance

Section 3: Impact of IPP on the IGV design

The major incentive for this project was to aid the IGVC efforts. The IGV team at Gannon needed a reliable method to navigate autonomously through an obstacle course crowded with traffic barrels. This project aimed to serve as a graduate thesis, as well as a solution for those IGV navigation needs.

One key aspect of the IGV competition is that the autonomous vehicles must meet restrictions on the size. The team from Gannon, like most teams at the competition, have opted to use the chassis and base of an electric wheelchair. An electric wheelchair base fits comfortably within the vehicle dimension rules and also allows for adequate room for electronics. Obviously, the compilation of electronics aboard the robot must include navigation, and therefore the navigation

electronics must be compact. The IGV team decided to utilize the NVIDIA Jetson TX2 module, shown in Figure 12(a) and the NVIDIA Jetson Xavier module, shown in Figure 12(b). These modules are compact AI computers that provide GPU performance with minimal power consumption. With that in mind, one had to ensure that the project is portable and able to be executed on the NVIDIA Jetson modules.

(a) (b)

Figure 12: (a) NVIDIA Jetson TX2 (b) NVIDIA Jetson Xavier

Testing the distance algorithm with the IGV

Figure 13 shows the testing arrangement. In order to test the distance algorithm in a more interactive setting, the Python program is ported onto a wheelchair-based platform. The wheelchair motors are programmed for the vehicle to follow the traffic barrel. The camera equipped vehicle follows the barrels and stops when the threshold distance setting is reached. Although it is impossible to show those results within a document, the testing was successful.

Figure 13: Python program test with a camera-equipped vehicle

Section 4: Engineering Project Activities supported by the IPP/IGV

The following are envisioned to benefit from the creation of the IPP/IGV.

- Courses with engineering project activities on the IPP/IGV
- Engineering curriculum development
- Engineering research activities
- Co-curricular activities

Courses with engineering project activities on the IPP

Table 1 lists some of the laboratory-based and project-based undergraduate, graduate, and technical elective (TE) courses offered across the Electrical Engineering (EE) or Embedded Systems (ES) programs at Gannon University. The TE courses are cross-listed with the graduate program (Master's degree in EE or ES). These courses can be adapted to include laboratory and project activities on the IPP. The theme of one or more laboratory and/or project experience is indicated. The details of each project activity will be determined by the instructor of the course in consultation with the technical documentation to be prepared.

Table 1: Courses with project activities on the IPP

Course	Year	Theme of the project activity on the IPP/IGV
Introduction to C/C++	Freshman	Programming the distance calculator
Digital Logic Design Lab	Freshman	PWM logic for motor speed control
Circuits I	Freshman	Powering electric vehicles with DC batteries
Circuits II	Sophomore	Understanding peripheral power management
Introduction to Microcontrollers	Sophomore	Cost/performance for various microcontrollers
Test and Measurement	Sophomore	Design interfaces to run diagnostics on the data
Electronics	Sophomore	Observing sensor/camera-based differences
Automatic Control Lab	Junior	Feedback control experiment
Project Experience	Junior	Project and team management skills to meet design specifications
Professional Seminar	Junior	Highlighting cross-discipline teamwork
Electric Drives Lab	Senior	Hardware-in-the-loop IGV controller design
Digital Image Processing (TE)	Graduate	Set up the IPP modules for different objects
Artificial Neural Networks (TE)	Graduate	Train and test different NN architectures
Applied Artificial Intelligence (TE)	Graduate	Design and test NN, OC and CA modules for a variety of objects
Python/MicroPython (TE)	Graduate	Integrate algorithms into Python architectures

Engineering curriculum development
The concepts required to understand the modules of the IPP span a set of EE and ES courses offered across the four year ABET accredited EE program at Gannon University. Therefore, for the student to gain a more comprehensive understanding of the laboratory and project experiences in the courses using the IPP, the EE program would have to align the curriculum to enable the students to see the integration of course content across a set of undergraduate courses, such as those listed in Table 1, each of which utilizes the IPP/IGV. This unification around the common theme helps build and strengthen student learning and retention of concepts.

Engineering research activities
The cross-listed and graduate level courses listed in Table 1 are suitable candidates for student research into alternate NN, OC, and CA designs for deep learning.

Co-curricular activities
This work in progress paper has been inspired by the motivation to enable the engineering students across our engineering disciplines to participate and compete in the IGVC. For instance, mechanical engineering students work on the design of the braking system of the IGV.

Section 5: Conclusions
The IPP is one of several possible engineering project platforms capable of supporting the vertical integration of courses through project activities in the courses taught across the four-year engineering curriculum. This will promote student retention of concepts as well as prepare them to be engineers with the breadth and depth of knowledge necessary to be effective and competitive in the workforce. The successful modular design and demonstration of the IPP,

together with its effective application to the redesign of the IGV, promises to deliver the following outcomes in the future.

- Active student participation in engineering projects over a wide range of complexity
- Encourages and instills *just-in-time, self-motivated,* and *self-directed* student learning
- Promotes team building
- Strengthens the communication and leadership skills in each student
- Gives each student *goal-oriented* and *purposeful* experiences

Section 6: Recommendations for Future Work

The IPP was designed and tested during the Fall 2020 semester. Future work, comprising the tasks and the expected timeline for completion, are shown in Table 2. During the Spring 2021 semester (January through May), the technical documentation of the IPP, which consists of the three modules (DCNN, OC, and CA), will serve as the reference manual for the operation of the IPP. In the Summer of 2021 (May through August), the laboratory experiments and project activities to be performed on the IPP in each of the courses listed in Table 1 will be identified, written, and tested. Specifically, the experiments and projects in each of the modules will be related to the course content. For instance, in the freshman course on Introduction to C/C++, the students will be tasked with writing and testing the program to compute the distance using the camera feed. In the course on micro-controllers, the students will evaluate, based on performance and cost, microcontrollers and sensors used to design the IGV. The course on neural networks (NN) will focus on deep learning models. During the Fall 2021 semester, the experiments and projects will be included in each course syllabus and mapped to the specific course outcome (CO) using the performance indicator (PI) for each ABET student outcome (SO).

Table 2: Tasks and timeline

Semester (Months)	Task	Details
Spring 2021 (Jan.-May)	Technical documentation of IPP	DCNN, OC, and CA modules
Summer 2021 (May-Aug.)	Prepare laboratory experiments and project activities on the IPP for the courses listed in Table 1 Redesign and test the IGV	Participate in the IGVC Develop and test the software, determine the hardware components required to interface with the courses and the IGV
Fall 2021 (Sept. – Dec.)	Include the experiments and projects in the courses and assess the learning outcomes	Map COs to the ABET SOs through the PIs for the undergraduate courses

Bibliography

[1] R. Sundaram, "Engineering Project Platform for Electrical and Computer Engineering Curriculum Integration," *Proceedings of the 2014 Annual ASEE conference & Exposition, Indianapolis, IN,* June 15-18, 2014. pp. 24.503.1-24.503.14, ISSN: 2153-5965; DOI: 10.18260/1-2-20394.

[2] K. Cheok et al., "The Intelligent Ground Vehicle Competition (IGVC): A Cutting-Edge Engineering Team Experience,", *Proceedings of the ASEE 2003 Annual Conference, Nashville, Tennessee, June 2003.*

[3] J. Lane et al., "Introduction: Intelligent ground vehicle competition (IGVC)," *Journal of Robotic Systems*, 07 July 2004; https://doi.org/10.1002/rob.20022.

[4] Cristian Iorga et al., "A Deep CNN Approach with Transfer Learning for Image Recognition," *2019 11th International Conference on Electronics, Computers and Artificial Intelligence (ECAI), Pitesti, Romania, June 27-29, 2019*; DOI: 10.1109/ECAI46879.2019.9042173.

[5] Mouna Aftf et al., "Indoor Object C1assification for Autonomous Navigation Assistance Based on Deep CNN Mode," *2019 IEEE International Symposium on Measurements & Networking (M&N), Catania, Italy, July 8-10, 2019*; DOI: 10.1109/IWMN.2019.8805042.

[6] P. Sharma et al., "Deep Convolutional Neural Network Design Approach for 3D Object Detection for Robotic Grasping," *2020 10th Annual Computing and Communication Workshop and Conference (CCWC)*, *Las Vegas, NV*, Jan. 6-8, 2020; DOI: 10.1109/CCWC47524.2020.9031186.

[7] Z-Q. Zhao et al., " Object Detection with Deep Learning: A Review,"arXiv:1807.05511v2 [cs.CV] 16 Apr 2019.

[8] N. Baker et al., "Deep convolutional networks do not classify based on global object shape," *PLoS Comput Biol 14(12): e1006613, 2018*. https://doi.org/10.1371/journal.pcbi.1006613.

[9] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov, "Scalable object detection using deep neural networks," arXiv:1312.2249v1 [cs.CV] 8 Dec 2013, in *CVPR* 2014.

[10] J. Redman et al., "You Only Look Once: Unified, Real-Time Object Detection," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* pp. 779-788, 2016.

[11] Juan Du, "Understanding of Object Detection Based on CNN Family and YOLO," IOP Conf. Series: Journal of Physics: Conf. Series 1004 (2018), DOI :10.1088/1742-6596/1004/1/012029.

[12] J. Redman and Ali Farhadi, "YOLO9000: Better, Faster, Stronger," arXiv:1612.08242v1 [cs.CV].

[13] K. Liu et al., "A Real-Time Method to Estimate Speed of Object Based on Object Detection and Optical Flow Calculation," IOP Conf. Series: Journal of Physics: Conf. Series 1004 (2018) DOI:10.1088/1742-6596/1004/1/012003.

[14] F. Particke et al., "Deep Learning for Real-Time Capable Object Detection and Localization on Mobile Platforms," IOP Conf. Series: Materials Science and Engineering 261 (2017) DOI:10.1088/1757-899X/261/1/012005.

[15] Y. Qi et al., "Vehicle Detection Under Unmanned Aerial Vehicle Based on Improved YOLOv3," 2019 12th International Congress on Image and Signal Processing, BioMedical Engineering, and Informatics (CISP-BMEI), Suzhou, China, 19-21 Oct. 2019. DOI: 10.1109/CISP-BMEI48845.2019.8966013.

[16] X. Zhang et al., "Vehicle Detection in the Aerial Infrared Images via an Improved Yolov3 Network," 2019 IEEE 4th International Conference on Signal and Image Processing (ICSIP), Wuxi, China, 19-21 July 2019. DOI: 10.1109/SIPROCESS.2019.8868430.

[17] S. Alhabshee et al., “Deep Learning Traffic Sign Recognition in Autonomous Vehicle,” 2020 IEEE Student Conference on Research and Development (SCOReD), Batu Pahat, Malaysia, 27-29 Sept. 2020. DOI: 10.1109/SCOReD50371.2020.9251034.

[18] S. Zhang et al., “Vehicle Detection in UAV Aerial Images Based on Improved YOLOv3,” 2020 IEEE International Conference on Networking, Sensing and Control (ICNSC), Nanjing, China, China, 30 Oct.-2 Nov. 2020, DOI: 10.1109/ICNSC48988.2020.9238059.

[19] J. Choi, “Uncertainty-based Object Detector for Autonomous Driving Embedded Platforms,” 2020 2nd IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS), Genova, Italy, 31 Aug.-2 Sept. 2020; DOI: 10.1109/AICAS48895.2020.9073907.

[20] D. Yoo et al., “AttentionNet: Aggregating Weak Directions for Accurate Object Detection,” arXiv:1506.07704v2 [cs.CV], in ICCV 2015.

[21] R. Girshick et al., “Rich feature hierarchies for accurate object detection and semantic segmentation,” arXiv:1311.2524v1 [cs.CV], in CVPR 2014.

[22] Y. Zhang et al., “Real-time vehicle detection and tracking in video based on faster R-CNN,” *IOP Conf. Series: Journal of Physics: Conf. Series 887 (2017)* DOI :10.1088/1742-6596/887/1/012068.

[23] K. He et al., “Mask R-CNN,” *2017 IEEE International Conference on Computer Vision (ICCV)*, 2017; DOI:10.1109/ICCV.2017.322.

[24] Y. Koo et al., “OpenCL-Darknet: An OpenCL Implementation for Object Detection,” *2018 IEEE International Conference on Big Data and Smart Computing (BigComp)*, Shanghai, China, Jan. 15-17, 2018; DOI: 10.1109/BigComp.2018.00112.

[25] N. McLellan et al., “Guidance for Specific Target Doors in Hallway using the Computer Vision for Autonomous Vehicles,” *2019 SoutheastCon, Huntsville, AL, USA. 11-14 April 2019.* DOI: 10.1109/SoutheastCon42311.2019.9020554.

[26] A. Mordvintsev and A. R. K., “Introduction to OpenCV-Python Tutorials,” *OpenCV*, 2013. [Online]. Available: https://docs.opencv.org/master/d0/de3/tutorial_py_intro.html. [Accessed: 29-Aug-2020].