Foundations of Autonomous Vehicles: A Curriculum Model for Developing Competencies in Artificial Intelligence and the Internet of Things for Grades 7–10

Elham Buxton[1], Elahe Javadi[2], Matthew Hagaman[2]
[1]University of Illinois at Springfield
[2]Illinois State University
esahe2@uis.edu, ejavadi@ilstu.edu, mthagam@ilstu.edu

Abstract

A few states (e.g., Georgia, and Florida) have initiated efforts to incorporate artificial intelligence outcomes in K12 education but others are still relying on informal spaces for learning and literacy in this area. In this manuscript, we share the curriculum and content of an informal effort focused on students in grades 7-10. We combined artificial intelligence competencies with Internet of Things skills to enable an experiential learning program that covers all Five Big Ideas in AI. In our one-week summer camp, students experimented with perceptions by working with vision, infrared, and ultrasonic sensors. They learned about representation through work with neural network playgrounds. Students engaged in supervised learning of an image processing model and used the model to control the actions of a robot car. Natural interactions and societal impacts were assessed as students observed the robot car's behavior. Results demonstrate that our curriculum was successful in achieving its objectives. Excluding the robot car kit, the curriculum was created using free platforms and tools. This program could be replicated in informal settings by any educator or collaborator with a computer science background. This paper describes our summer camp curriculum, its components and their implementation, the lessons learned, and potential future enhancements.

Introduction

Artificial Intelligence (AI) and the Internet of Things (IoT) are key elements in the current wave of digital transformation (Bodrožić and S. Adler 2022). As technology is transformed, new career options are being created and AI literacy is becoming an essential part of technological literacy (Ng et al. 2021). A few states (e.g., Georgia and Florida) are working to incorporate AI learning outcomes in their K12 curricula[1]. All states and all nations must tackle the same change in K12 education (Song et al. 2023). It is essential for K12 education to adequately prepare students to be citizens in an AI-driven society, whether through in-school instruction or through educational programming in informal settings (Ali et al. 2021; Lao 2020). Similarly, the Internet of Things is becoming an integral part of citizens' lives as more people adopt home security or home assistant devices. IoT devices are now a key component of many machine learning pipelines (Kreuzberger, Kühl, and Hirschl 2023) such as those used in agriculture to collect crop data (Esau et al. 2023) and those placed on robots to get feedback from the physical world (Morenville et al. 2022). IoT and AI are related because AI perception is often achieved through the use of sensors and IoT devices play a key role in robotics and beyond (Touretzky, Gardner-McCune, and Seehorn 2022; Mohammadi et al. 2018).

Our curriculum integrates AI and IoT competencies and it enables experiential learning through active experimentation and analysis of AI's five big ideas: perception, representation and reasoning, learning, natural interaction, and societal impact (Touretzky, Gardner-McCune, and Seehorn 2022). It was developed based on AI for K12 literature and following Kolb's principles of experiential learning (Kolb 1984) . Program participants engaged in concrete experiments with IoT and AI tools and techniques, first separately, then together. Autonomous vehicles were used as a vessel through which students engaged in synthesizing the two sets of competencies and engaged in abstract conceptualizations. Autonomous vehicles offer a variety of AI-related challenges and tasks (Tang et al. 2018; Peters et al. 2019); we followed Tang and colleagues (Tang et al. 2018) to create a few foundational and simplified modules that can be accomplished during a one-week program. In the last part of the program, students engaged in active experimentation and showcased a diverse set of approaches to how the two competencies can make a unique type of behavior in autonomous vehicles. Throughout the experience, students engaged in reflective observation with the help of camp facilitators. The overarching goal of this effort is to broaden education in AI, enhance middle and high-school students' self-efficacy, and to foster career aspirations by exposing students to the field at both theoretical and applied levels (Weinberg, Basile, and Albright 2011).

Related Work

Researchers in engineering, computing, and education have tackled the topic of AI literacy in a variety of ways and have created many tools, platforms, and curricula to support this pursuit. Touretzky and Gardner-McCune, for instance, made it easy for beginners to experiment with computer vi-

[1]https://www.gadoe.org/Curriculum-Instruction-and-Assessment/CTAE/Documents/Artificial-Intelligence-Applications.pdf

sion by creating Calypso, a software product where users interact with a vision-enabled robot called Cozmo (Touretzky and Gardner-McCune 2018). They emphasized the importance of including perception as one of the five ideas in AI education. Anton and colleagues (Anton, Behne, and Teuteberg 2020) reviewed practical and research literature, as well as thousands of AI job postings with the aim of defining the boundaries and scope of AI competencies. Ng and colleagues (Ng et al. 2021) reviewed the literature to derive the technological, teaching and learning contents as well as the pedagogical elements of AI literacy. They emphasized that AI literacy is a matter of concern for the general student population (and not just computer science students), and that students must earn competencies that would allow them to perform not merely at the usage level but at the decision-making and problem-solving levels.

AI curriculum development efforts will benefit from development of community-built concept inventories, objectives, and essential competencies (Schleiss et al. 2022). Pioneering organizations and communities such as AI4K12[2] (other examples: AI4All[3], and AI4GA[4]) are becoming de facto mechanisms through which educators share resources. The curriculum presented here was also developed using a number of resources and tools available through A4K12. The one-week curriculum described here aims to introduce the five big ideas of AI and explore their interrelation by combining IoT and AI competencies. The building blocks are introduced and dissected through an experiential learning journey (Weinberg, Basile, and Albright 2011; Broll and Grover 2023). Students not only train and refine their computer vision models but also observe their models' concrete application in their smart cars and evaluate their models' behavior in the physical environment. Unlike most one-week camps which typically focus on either AI or IoT competencies, we were able to synthesize the two topics in the context of autonomous vehicles using available resources provided by other researchers and educators (Payne 2019).

During the one-week experiential journey, students had to tackle issues such as model bias, data distribution shift, real-time performance limitations (e.g., speed of inference, image quality, background noise, connection issues), and adverse impacts of perception-action mismatches. For instance, while a misclassified image or slow speed of inference may not render themselves as important factors when students work with web-based tools such as Google Teachable Machine[5], students can directly observe the consequence of these issues through their interaction with hardware and in using model's output for making decisions about the car movement.

Setting the autonomous vehicles as the context for this curriculum allowed discussion of natural interaction and societal impacts in a more concrete way for these young learners (Touretzky, Gardner-McCune, and Seehorn 2022). All humans interact with cars and we all contribute to and are impacted by the conditions and safety (or lack thereof) of the roads that we are sharing. Autonomous vehicles will interact with pedestrians, bikers & other stakeholders (Payne 2019) in the transit ecosystem (Weinberg, Basile, and Albright 2011). Autonomous vehicles also are conducive to deep discussions around algorithmic accountability, equity, and ethics (Pang et al. 2023). Because the interactions between human and autonomous vehicles are not human-to-human interactions (one driver to another, pedestrian-to-driver), ethical dilemmas such as those described in Trolley Problem[6] must be reconsidered, discussed, and pondered in a new way. This curriculum's primary purpose is to harness the intrigue of autonomous vehicles to foster AI and IoT literacy among 7-10 grade students.

Competency	Objective
IoT	Experiment with micro-controllers and sensors; Identify major sensors used in a simple model of autonomous vehicles
AI	Explore five big areas of AI and identify facets of autonomous vehicles related to each of the five areas; Experiment with training and performance evaluation of computer vision models
IoT+AI	Synthesize perception, representation, learning, and natural interactions to design a specific behavior in the smart car; Assess societal impacts of the design for stakeholders and their interactions with the car

Table 1: Autonomous vehicle camp competency areas and related objectives

Foundations of Autonomous Vehicles Curriculum

Our curriculum consists of three major modules: (1) IoT, (2) AI, and (3) Integration of AI and IoT. The corresponding objectives for each of these three components are listed in Table 1.

Each of the main five program-level objectives was achieved through a set of activities that involved individual, pair/group, and cohort-level elements and followed Kolb's model (Kolb 1984). Details of each activity and related resources are shared in their respective sections; however, the overall camp was made possible through the use of a few essential resources listed in Table 2.

[2]https://ai4k12.org/
[3]https://ai-4-all.org/
[4]https://ai4ga.org/
[5]https://teachablemachine.withgoogle.com/
[6]https://www.britannica.com/topic/trolley-problem
[7]https://www.elegoo.com/products/elegoo-smart-robot-car-kit-v-4-0
[8]https://pixspy.com/
[9]https://docs.python.org/3/library/socket.html
[10]https://github.com/eloquentarduino/everywhereml

Competency	Resources
IoT	Arduino Microcontroller; ESP 32 camera dev module (with Wi-Fi capability), ultrasonic sensor, and infrared sensors; Commercial product used: Elegoo Smart Car v.4.0[7]
AI	Python IDE; printed images of traffic signs for the computer vision model; 20×20 blank pixel art paper for image representation; Platforms used: Google Teachable Machine, Pix Spy image inspection tool[8]
IoT+AI	Python IDE, keras, socket[9], tensorflow, and everywhereml libraries[10]

Table 2: Resources used for the curriculum

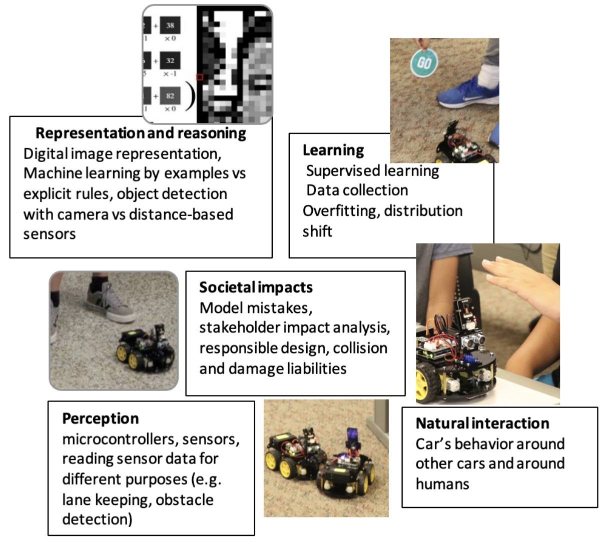

Figure 1: Curriculum components covering five big ideas of AI

Target Audience

We designed this curriculum for grades 7-10. The age group was selected to increase the potential for impacting students' motivation to pursue careers involving computational thinking, especially those focused on AI. The curriculum was implemented in a week-long all-day summer camp with a total instruction time of approximately 20 hours.

Prerequisite Knowledge

No prerequisite knowledge is assumed for this curriculum. However, students may require assistance from an instructor with a computer science background to perform some of the curriculum activities (particularly those that involve modifying a given code segment).

Curriculum Activities and their Alignment with Kolb's Model and AI Big Ideas

Table 3 shows activities performed in each of the three modules of the camp: IoT, AI, and IoT+AI and their alignments to Kolb's phases of experiential learning cycle (Kolb 1984): (1) concrete experiment, (2) reflective observations, (3) abstract conceptualization, and (4) active experimentation. In this table, *I* denotes an individual activity, *G* denotes a pair or group activity, and *C* denotes a cohort activity. Figure 1 shows the components of the curriculum that cover the five big ideas of AI. In the following sections, we share details of the curriculum activities related to each learning modules in Table 3.

IoT Module

The camp curriculum starts with the IoT module; students assemble their robot cars, get acquainted with the IoT devices, experiment with sending and receiving data to and from the environment, and learn to process sensor data using a microcontroller.

While we have used the commercial product ELEGOO UNO R3 Project Smart Robot Car Kit V 4.0 (With Camera) in our camp, we anticipate that educators with more experience with controllers, sensors, and robot design could build an in-house robot to follow this curriculum. Additionally, this curriculum can be easily adapted for use in maker-space venues or robotics clubs where hardware and sensor supplies are available. The choice to use a commercial smart car was made based on three factors: (1) the availability of documentation and resources to enable learning beyond camp and the potential of cascading impact beyond the camp scope; (2) the affordances of the informal space that was used for the camp (a computer lab, not a maker space with tools); and (3) low cost of ownership of the commercially available robot car kit (ranging from $55-$80 depending on purchase time). Our pre-program research and post-program assessment corroborates that the selected commercial kit was indeed suitable for beginner-level robotics learning. The Arduino-based design allowed free access to the programming IDE at home. The smart car has a Wi-Fi-enabled ESP32 camera development module mounted on the car which supports AI competencies described in the next sections. The ESP32 camera development module is an inexpensive vision sensor that allows easy replacement when necessary and more liberal experimentation beyond camp.

Robot Car Assembly

The car kit is designed for ages 12+ and comes with an easy-to-follow manual for assembly. Students can follow the paper manual or use ElegooKit app (which is available to install on both mobile devices and PCs) for visual step-by-step instructions.

Exploring Microcontroller-Sensor Interaction

During assembly, instructors are recommended to explain the role of the microcontroller and the function of sensors in the robot car. Students are then allowed to play with their assembled cars, exploring features such as obstacle avoidance,

module	Activity	Experiential Learning Phase	Activity Level	Duration
IoT	Robot car assembly	concrete experimentation	I	0.5 day
	Exploring Microcontroller-Sensor Interaction & connections	Reflective observation	G, C	1-2 hours
	Programming microcontroller using both graphical interface and Arduino IDE	Active experimentation	I, G, C	0.5 day
AI	Discussion on Computer Vision in Autonomous Vehicles	Concrete Experimentation	I, C	1 hour
	Exploring digital representation of Images in Computers & Motivation for Machine Learning	Reflective Observation, Abstract Conceptualization	G, C	1-2 hours
	Training computer vision models to recognize hand-gesture	Active experimentation	I, G, C	0.5 day
IoT+AI	Controlling the Robot Car with hand gesture and traffic signs	Active experimentation	I, G, C	0.5 day
	Integrating IoT, AI and Ethics in a culminating project	Reflective Observation, Abstract Conceptualization, Active experimentation	I, G, C	1 day

Table 3: Curriculum activities and their corresponding phases of experiential learning

line tracking, and auto follow modes. The car can be controlled through an infrared remote control or via ElegooKit app. After assembling the car, students are engaged in discovery and reflection of how the microcontroller and sensors interact to create a certain behavior in their autonomous vehicle, such as obstacle avoidance. This can be achieved by first using an intuitive drag-and-drop graphical programming interface and then transitioning to lower level coding via Arduino IDE.

Programming the Microcontroller via a Graphical Interface

The ElgooKit App provides a graphical programming interface for programming the Elegoo UNO R3 board to control the car. This programming requires the device running the App to be connected to the Wi-Fi module of the ESP32 board. The graphical programming interface includes three groups of statements: (1) Motion: instructions that control the movement of the car. (2) Control: program control structures, including conditional statements and loops. (3) Sensing: statements that require input from car sensors, including Boolean expressions for determining if the car was picked up or if an obstacle has been detected, as well as taking measurements such as the distance to an obstacle or the values of infrared sensors. Students are provided with a description of each statement group and are asked to imagine a movement pattern of their choice. They then are asked to create a sequence of statements from the three groups to execute the pattern. Figure 2 illustrates the graphical programming interface together with an example program. This program performs an *obstacle avoidance* operation. It first resets the micro servo, which holds both the ultrasonic sensor and the ESP32 camera, to a 90-degree angle. It then moves the car forward until it detects an obstacle within a 50cm distance. At that point, the car stops to check distances to obstacles on both its left and right sides and turns the car in the direction where the obstacle is furthest away, before continuing

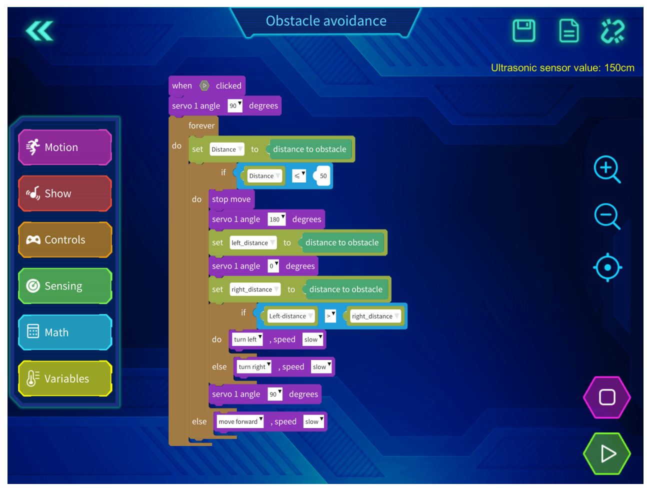

Figure 2: An example program in Elegoo Graphical Programming Interface. This example program performs obstacle avoidance.

forward.

Programming the Microcontroller with Arduino IDE

After exploring the car kit capabilities and programming via a drag-and-drop visual interface, students can be introduced to programming the microcontroller using lower level coding and Arduino IDE. The goal is to understand how information is transmitted from the sensors to the microcontroller board via input pins, how the board processes that information and how it subsequently sends information to the motors using output pins. To achieve this goal, students are given three examples with incremental complexity:

- *Blink example*: Arduino IDE's example turns a LED light on and off every second[11]. Students learn to use the proper port and board configuration to successfully upload and run the code, and can modify the script's delay to see its impact.

- *Servo sweep example*: built-in *servo sweep* example[12] that sweeps the shaft of the microservo back and forth across 180 degrees. Students need to find the pin microservo is connected to on the microcontroller board and modify the step size of the angle to control the servo movement.
- *Distance measure example*: Once students get familiar with how information is transmitted in and out of the microcontroller, they can progress to a slightly more complex example, such as the code that measures distance to an obstacle using the ultrasonic sensor. Students then are tasked with enhancing this example to turn on the board's LED light or rotate the microservo if the distance to an obstacle is less than a certain threshold.

These activities help students make connections between each successive step, providing a seamless transition to the next activity which is enhancing the robot car with computer vision.

AI Module

The obstacle detection activity in the IoT module naturally paves the way for introducing computer vision in autonomous vehicles.

Discussion on Computer Vision in Autonomous Vehicles

To begin a conversation about computer vision, instructors can pose questions such as *can an ultrasonic sensor be used to differentiate between different obstacles (human, wall, chair, etc.) in front of the car?* and *is there another sensor in the robot car capable of perceiving such information from the environment?*. This discussion prompts students to think about how computers perceive and interpret visual data about the world around them. A short introductory video on AI and computer vision in the context of autonomous vehicles, such as the one developed by code.org[13] can help answer some of these questions for students.

Digital Image Representation & AI Motivation

In the next part of the curriculum, we introduce a simplified computer vision problem: enabling the robot car to recognize different hand gestures captured by its camera. The initial step is to understand how an image of a hand gesture is stored as a grid of RGB integer values. Students are instructed to connect to the webserver of the ESP32 camera of their robot car and capture an image of a hand gesture. They then upload the image to an interactive web-based tools such as Pix Spy[14], where they can click on various parts of the image to see the corresponding RGB pixel values.

To motivate machine learning as opposed to rule-based approach for computer vision, instructors can hand out blank 20×20 pixel art papers to the students who will then form pairs. In each pair, one student draws a black-and-white pixel representation of a hand gesture. The other student tries to formulate a set of rules that describes the hand gesture based on the position of the black pixels. The first student then subtly alters the pixel art to still represent the same hand gesture but violate these rules. Through a few rounds of this exercise students grasp the inherent complexity of the image recognition task for machines and gain an understanding of the advantages of teaching machines to recognize objects in images through examples rather than through explicit rules.

Training a Computer Vision Model for Hand Gesture Recognition

At this point, students are ready to work with visual tools such as Google Teachable Machine to train a machine learning model with example images of different hand gestures that will later be used to control the robot car movement.

Training Image Classification Using Computer's Camera Students first use the computer's camera to capture images of different hand gestures (e.g., stop, go, left, right, and none [i.e., no hand gesture in the picture]). Students are encouraged to collect a diverse and inclusive set of training images with various skin tones and different backgrounds. After multiple rounds of model training and evaluation, once students are satisfied with the accuracy of their model, they connect to the ESP32 Wi-Fi module on the car, open the camera web interface to capture an image and upload it to Teachable Machine for testing and inference. Students may observe that the model does not detect hand gestures from the images captured by the car camera as accurately as it does with those captured by the computer's camera. This step helps students grasp the concept of *distribution shift* (Moreno-Torres et al. 2012) where a model's performance diminishes as a result of deviation between the training data and inference data. This motivates the next step of retraining the model on hand gesture images collected using the car camera.

Training Image Classification Using Robot Car Camera Students can use *everywhereml*[15] or other similar libraries in Python to collect images directly from the live feed of the camera mounted on the robot car. As no coding background is assumed, a sample code such as the one in Figure 3 can be provided to students. This sample code connects to the ESP32 video streaming web server, uses *everywhereml* to collect live video frames over HTTP for each class of hand gesture then stores the frames in a sub-folder with the same name as the respective hand gesture. Students subsequently upload these images to train a new model on Google Teachable Machine and observe that it has a higher accuracy at in-

[11] https://docs.arduino.cc/built-in-examples/basics/Blink

[12] https://wiki-content.arduino.cc/en/Tutorial/LibraryExamples/Sweep

[13] https://studio.code.org/s/how-ai-works-2023/lessons/2

[14] https://pixspy.com/

[15] https://eloquentarduino.com/

```
#import everywhereml library for collecting images
from everywhereml.data import ImageDataset
from everywhereml.data.collect import MjpegCollector

'''
specify the folder where you want to
save images for each hand gesture
'''
base_folder = 'car_images'

#Specify the the camera streaming web server URL
IP_ADDRESS_OF_ESP_STREAM = 'http://192.168.4.1:81/stream'

#Initialize MjpegCollector
mjpeg_collector = MjpegCollector(address=IP_ADDRESS_OF_ESP)

#start collecting images.
image_dataset = mjpeg_collector.collect_many_classes(
        dataset_name='Dataset',
        base_folder=base_folder,
        ''' the duration for capturing images,
            change duration to a higher value
            if you want to capture more images
        '''
        duration=50
    )
This is an interactive data capturing procedure.
Keep in mind that when you enter a class name,
the capturing will start in 2 seconds,
Which class are you going to capture? (leave empty to exit)  Go
100%|██████████| 99.83529376983651/100 [00:49<00:00,  2.00it/s]
100%|██████████| 100.00691556930549/100 [00:49<00:00,  2.00it/s]
INFO:root:Captured 5328 images
Is this class ok? (y|n)  y
Which class are you going to capture? (leave empty to exit)
```

Figure 3: A sample code snippet for collecting images from the robot car's camera using *everywhereml* library.

ference time compared to the model they previously trained using the live feed from the computer camera.

Integrative IoT+AI Module

Google Teachable Machine is a great tool for allowing younger students to explore computer vision by training their own image classification models; however, to fully recognize the application of the models they've trained, students need to see how it interacts with the physical environment.

Establishing Communication between the AI Model and the Microcontroller

The goal in this activity is to make the motor on the robot car respond to different hand gestures recognized by the trained model in real time. To accomplish this goal, students can download the model they trained in Google Teachable as a Keras model, then load the model in python and use it for inference. The specific steps are outlined below. Instructors provide the code snippet used in each of these steps to students and explain its overall functionality.

1. Capture an image from the car camera. A simple function, such as the one shown in Figure 4 can be provided to students for capturing a single image from the car camera using *everywhereml* or other similar libraries.
2. Load the image classification model from the disk and use it to predict the hand gesture in the image captured (Figure 5).

```
from PIL import Image, ImageOps
import numpy as np
def get_image(mjpeg_collector):
    """
    Captures an image using a MjpegCollector
    and prepares it for the image classificaiton model
    Parameters:
        mjpegCollector: is an MjpegCollector object
    Returns:
        processed image
    """
    #capture a single image from the camera
    test_sample = mjpeg_collector.collect_by_samples(num_samples=1)
    #image capturesd is in raw bytes convert to RGB
    image = Image.open(io.BytesIO(test_sample[0])).convert("RGB")
    #display the image
    image.show()
    '''
    the following code segment is copied from google teachable
    it resizes and normalizes an image for prediction
    '''
    data = np.ndarray(shape=(1, 224, 224, 3), dtype=np.float32)
    # resizing the image then cropping from the center
    size = (224, 224)
    image = ImageOps.fit(image, size, Image.Resampling.LANCZOS)
    # turn the image into a numpy array
    image_array = np.asarray(image)
    # Normalize the image
    normalized_image_array = (image_array.astype(np.float32) / 127.5) - 1
    # Load the image into the array
    data[0] = normalized_image_array

    return data
```

Figure 4: A sample code snippet for a function that captures a single image from the car camera, displays it, and prepares it for the model.

3. Use the socket library in python to connect to the camera webserver and send a command from python to Arduino via Wi-Fi to control the motor. This requires that the Wi-Fi module of ESP32 be set to pass-through mode allowing the data it receives to be transferred to the serial port. In Elegoo's robot car, the serial port of the ESP32 is directly connected to the serial port of the Arduino allowing the data to be passed to Arduino for further processing. Elegoo uses a fixed JSON data format to communicate with the board via Wi-Fi when the ElegooKit app is used to control the motor. The factory code of Arduino board reads the JSON string in the serial buffer, parses it, and executes the command. For instance, when the joystick on the app is moved forward, Elegoo sends the command { "N": 102 , "D1": 1 , "D2": parameter2 }, where "N" specifies the type of command (102 represents *joystick*), D1 is the direction to move the motors (1 represents *forward*) and D2 is the speed. This command is passed through the Wi-Fi module of ESP32 to Arduino's serial input where it is parsed and used to subsequently move the motors. Figure 6 provides a sample code snippet for step 3. We used the forward command as an example but encouraged students to experiment with different commands provided in the communication protocol in Elegoo documentation[16].
4. Put the above three steps together. A sample code snippet, such as the one shown in Figure 7, can be provided to students to combine the above steps. This sample code continuously captures an image from the car

[16]https://www.elegoo.com/blogs/arduino-projects/elegoo-smart-robot-car-kit-v4-0-tutorial

```python
#import required libraries
from tensorflow import keras
from keras.models import load_model

#Specify the the camera streaming web server URL
IP_ADDRESS_OF_ESP_STREAM = 'http://192.168.4.1:81/stream'

#Initialize MjpegCollector
mjpeg_collector = MjpegCollector(address=IP_ADDRESS_OF_ESP_STREAM)

#giving a heads up
print("Capturing image in 2 seconds...")
time.sleep(2)

#capture a single image from camera live feed and preprocess for model
image=get_image(mjpeg_collector)

# Load the Keras model downloaded from Teachable Machines
model = load_model("keras_Model.h5", compile=False)

# Load the labels downloaded from Teachable Machines
class_names = open("labels.txt", "r").readlines()

# use model to predict the gesture for the image captured by camera
prediction = model.predict(image)
index = np.argmax(prediction)
class_name = class_names[index]
confidence_score = prediction[0][index]

# Print prediction and confidence score
print(f"""model Predicted: {class_name[2:]}
      with confidence score{confidence_score}""")
```

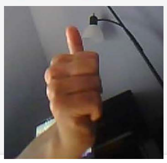

Figure 5: A sample code snippet for getting the model's prediction on an image captured from the car camera. The section on loading and using the model is adapted from Google Teachable Machines.

```python
import socket
import time
#specify the ip address of the camera server
IP_ADDRESS_OF_ESP="192.168.4.1"

'''specify the port to receive communication
(Use port 100 for Elegoo robot car kit)'''
PORT=100

#start the connection
s=socket.socket()
s.connect((IP_ADDRESS_OF_ESP,PORT))

'''send forward command
See Communication protocol in Elegoo Robot Car'''
s.sendall(b'{"N":102,"D1":1,"D2":20}')

#wait for one second
time.sleep(1)

# send a command to stop the car
s.sendall(b'{"N":100}')

#close the connection
s.close()
```

Figure 6: A sample code snippet for communicating with motor using Elegoo communication protocol.

camera, passes it to the model for prediction, and sends a command through Wi-Fi to Arduino to move the motors based on the output of the model.

Integrating IoT, AI, and Ethics in Student Projects

On the last day of the camp, students are given the opportunity to work in a small group to create a project that builds on the three components they have learned: IoT, AI, and IoT+AI. This group work serves as a culminating experience, allowing students to think about other scenarios where they can utilize computer vision to enhance their robot car capabilities. Depending on the specific behavior intended for training their robot car, this step may require modifying sample code snippets from previous camp activities and will be possible with instructors assistance. Based on our camp implementation experience, the majority of students were able to to pinpoint sections in the code snippets they needed to modify to adapt it to their intended behavior. They effectively formulated algorithmic steps for these changes, which were subsequently translated into Python code with the help of instructors. Camp organizers can provide additional materials such as printed pictures of traffic signs to encourage further reflection on the real-world application of what students have learned. In addition, students can discuss ethical issues and stakeholder impact assessment (Payne 2019), to reflect on societal impact of the autonomous vehicles in general as well as those unique for their designed vehicle behavior. Students can be encouraged to identify various stakeholders (e.g., drivers, pedestrians, manufacturers, legislators, etc.) who would be affected by their project if implemented in real-world autonomous vehicles, examining how their values might overlap or conflict. Students communicated this information to their peers as a part of an end-of-camp showcase beginning with 30-second lightning talks and culminating in more in-depth demonstrations.

Implementation

We piloted the curriculum activities described here with two cohorts consisting of eighteen and eleven students from grades 7-10 in two different regions of central Illinois. To encourage inclusion and diversity, we used camp vouchers to recruit students from undeserved communities. The makeup of the students was approximately 35% Asian, 35% White, and 30% African American. At the end of the 5-day workshop, 90% of the students reported positive feelings about the program in relation to *building robots*, *learning to do programming*, and *experimenting with codes* which are indicators of experiential learning or learning by doing. Students took divergent paths when combining their AI and IoT skills and developing a novel project. Some students refined their machine learning models using gesture detection while others trained new models based on traffic signs (e.g., stop, go, curve left, curve right). One pair of students added an additional servo to allow the car to control a pen-spring cannon and another group created an *anti-social robot* that used the camera to detect human obstacles and change its direction. A group of three students used printed traffic signs to create an obstacle course for the robot car. Each project proved successful, with every group completing their chosen project.

```
#specify the ip address of the camera server
IP_ADDRESS_OF_ESP="192.168.4.1"
'''specify the port to receive communication
(port 100 for Elegoo robot car kit)'''
PORT=100

while True:
    print("capturing image...")
    #capture image from camera and preprocess
    image=get_image(mjpeg_collector)
    #connect to the camera webserver
    s=socket.socket()
    s.connect((IP_ADDRESS_OF_ESP,PORT))
    # use model to predict the gesture in the image
    prediction = model.predict(image)
    index = np.argmax(prediction)
    class_name = class_names[index][2:].strip()
    confidence_score = prediction[0][index]

    # Print prediction and confidence score
    print(f"""model Predicted: {class_name}
        with confidence score{confidence_score}""")
    if class_name=="Go":
        #Send forward command
        s.sendall(b'{"N":102,"D1":1,"D2":20}')
    elif class_name=="Left":
        #Send Left command
        s.sendall(b'{"N":102,"D1":3,"D2":20}')
    elif class_name=="Right":
        #Send Right command
        s.sendall(b'{"N":102,"D1":4,"D2":20}')
    else:
        #send Stop command
        s.sendall(b'{"N":100}')
     #wait for two second before capturing another image
    time.sleep(2)
    #close the connection
    s.close()
```

Figure 7: A sample code snippet that captures an image from the car camera every second, uses the model to recognize the hand gesture in the image, and sends an appropriate command to the motor. This code snippet assumes that the model is trained to recognize four hand gestures (*stop*, *go*, *left*, and *right*).

Conclusion and Future Enhancements

In this paper we presented a curriculum model for a week-long summer camp targeting grade 7-10 students. This curriculum integrates AI and IoT competencies with a focus on application of computer vision in autonomous vehicles covering all five big ideas of AI. The curriculum uses freely available tools and resources and can be implemented in informal settings with the help of an instructor with CS background. Our pilot implementation of this curriculum successfully achieved its learning objectives and a number of students expressed interest in continuing to experiment with their robot cars beyond the camp.

For further iterations of this work, we aim to refine the existing curriculum and pilot it with a larger cohort of students. The goal is to conduct user-based studies to examine the curriculum's impact on students' situational motivation, AI literacy, and STEM career aspiration using standard survey instruments (Guay, Vallerand, and Blanchard 2000; Weinberg, Basile, and Albright 2011).

Evolving this curricular module can involve delineating most appropriate subsets for different grade levels. We realize that the students in 7th and 10th grade are equipped with distinct levels of mathematics and algebra. This module did not require or assume specific proficiency level in mathematics. However, future specialized modules can leverage capabilities of students in higher grades to more deeply explore concepts such as convolution, back propagation, and object-distance calculation (in ultrasonic module).

Other refinements to the curriculum can involve enhancing the Integrative IoT+AI Module with real-time inference. Currently, the curriculum relies on Wi-Fi connection for both receiving images from the camera and transmitting motor commands to the microcontroller. While this approach provides a near real-time inference experience for students, it could be hindered by the speed of Wi-Fi connection. This process can be improved by deploying the model directly onto the the ESP32 module creating a real-time, low-latency inference that more closely resembles the operation of actual autonomous vehicles.

Acknowledgments

This material is based upon work supported by the Illinois Innovation Network under Contract Number CN-00078253. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the UNIVERSITY or Illinois Innovation Network.

References

Ali, S.; DiPaola, D.; Lee, I.; Sindato, V.; Kim, G.; Blumofe, R.; and Breazeal, C. 2021. Children as creators, thinkers and citizens in an AI-driven future. *Computers and Education: Artificial Intelligence*, 2: 100040.

Anton, E.; Behne, A.; and Teuteberg, F. 2020. The humans behind Artificial Intelligence–An operationalisation of AI competencies. In *n Proceedings of the 28th European Conference on Information Systems (ECIS)*.

Bodrožić, Z.; and S. Adler, P. 2022. Alternative futures for the digital transformation: A macro-level Schumpeterian perspective. *Organization Science*, 33(1): 105–125.

Broll, B.; and Grover, S. 2023. Beyond Black-Boxes: Teaching Complex Machine Learning Ideas through Scaffolded Interactive Activities. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(13): 15990–15998.

Esau, T. J.; Hennessy, P. J.; MacEachern, C. B.; Farooque, A. A.; Zaman, Q. U.; and Schumann, A. W. 2023. Artificial intelligence and deep learning applications for agriculture. In *Precision Agriculture*, 141–167. Elsevier.

Guay, F.; Vallerand, R. J.; and Blanchard, C. 2000. On the assessment of situational intrinsic and extrinsic motivation: The Situational Motivation Scale (SIMS). *Motivation and emotion*, 24: 175–213.

Kolb, D. 1984. *Experiential learning: experience as the source of learning and development*. Englewood Cliffs, NJ: Prentice Hall.

Kreuzberger, D.; Kühl, N.; and Hirschl, S. 2023. Machine learning operations (mlops): Overview, definition, and architecture. *IEEE Access*.

Lao, N. 2020. *Reorienting machine learning education towards tinkerers and ML-engaged citizens*. Ph.D. thesis, Massachusetts Institute of Technology Cambridge, MA, USA.

Mohammadi, M.; Al-Fuqaha, A.; Guizani, M.; and Oh, J.-S. 2018. Semisupervised Deep Reinforcement Learning in Support of IoT and Smart City Services. *IEEE Internet of Things Journal*, 5(2): 624–635.

Moreno-Torres, J. G.; Raeder, T.; Alaiz-Rodríguez, R.; Chawla, N. V.; and Herrera, F. 2012. A unifying view on dataset shift in classification. *Pattern Recognition*, 45(1): 521–530.

Morenville, A.; Vermeulen, L.; Schaus, P.; and Nijssen, S. 2022. Simultaneous localization and mapping and autonomous navigation on SPOT robot from Boston Dynamics. https://dial.uclouvain.be/memoire/ucl/object/thesis:35688. Accessed: 2024-01-04.

Ng, D. T. K.; Leung, J. K. L.; Chu, S. K. W.; and Qiao, M. S. 2021. Conceptualizing AI literacy: An exploratory review. *Computers and Education: Artificial Intelligence*, 2: 100041.

Pang, H. N.; Parks, R.; Breazeal, C.; and Abelson, H. 2023. How Can I Code AI Responsibly?: The Effect of Computational Action on K-12 Students Learning and Creating Socially Responsible AI. In *Pang, H Nicole and Parks, Robert and Breazeal, Cynthia and Abelson, Hal*.

Payne, B. H. 2019. An ethics of artificial intelligence curriculum for middle school students. *MIT Media Lab Personal Robots Group. Retrieved Oct*, 10: 2019.

Peters, D.; Bastiaan, J.; Pimentel, J. R.; and Zadeh, M. 2019. The AutoDrive Challenge: Autonomous Vehicles Education and Training Issues. *ASEE Peer*.

Schleiss, J.; Bieber, M.; Manukjan, A.; Kellner, L.; and Stober, S. 2022. An interdisciplinary competence profile for AI in engineering. In *Towards a new future in engineering education, new scenarios that european alliances of tech universities open up*, 1601–1609. Universitat Politècnica de Catalunya.

Song, J.; Yu, J.; Yan, L.; Zhang, L.; Liu, B.; Zhang, Y.; and Lu, Y. 2023. Develop AI Teaching and Learning Resources for Compulsory Education in China. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 16033–16039.

Tang, J.; Shaoshan, L.; Pei, S.; Zuckerman, S.; Chen, L.; Shi, W.; and Gaudiot, J.-L. 2018. Teaching autonomous driving using a modular and integrated approach. In *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, volume 1, 361–366. IEEE.

Touretzky, D.; Gardner-McCune, C.; and Seehorn, D. 2022. Machine learning and the five big ideas in AI. *International Journal of Artificial Intelligence in Education*, 1–34.

Touretzky, D. S.; and Gardner-McCune, C. 2018. Calypso for Cozmo: Robotic AI for Everyone (Abstract Only). In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, SIGCSE ’18, 1110. New York, NY, USA: Association for Computing Machinery. ISBN 9781450351034.

Weinberg, A. E.; Basile, C. G.; and Albright, L. 2011. The effect of an experiential learning program on middle school students’ motivation toward mathematics and science. *RMLE Online*, 35(3): 1–12.