Artificial Intelligence Powered Object Capture

Gabriel Rodriguez
Department of Physical Sciences
Embry-Riddle Aeronautical University
Daytona Beach, Florida
rodrig43@my.erau.edu

Jose Castelblanco
Department of Physical Sciences
Embry-Riddle Aeronautical University
Daytona Beach, Florida
castej10@my.erau.edu

***Abstract*—Artificial intelligence (AI) and deep learning hold vast potential across diverse domains. This project harnesses AI for object detection and capture, focusing on free-falling objects using an omnidirectional vehicle equipped with a singular camera. The vehicle identifies the object's position and adjusts its trajectory for capture. In unrestricted testing, the vehicle exhibited a success rate of 48%. While this level of success serves as a proof of concept, hardware limitations hindered further optimization. The project demonstrates the feasibility of AI-driven object capture and highlights opportunities for broader applications, including space debris capture, industrial diagnostics, and defense systems. Despite hardware constraints, the study lays groundwork for future developments in AI-powered engineering solutions.**

I. Introduction

This research project aimed to tap into the capabilities of Artificial Intelligence (AI). The primary focus was on exploring AI's capabilities in detecting and capturing objects. The result culminated in the use of AI for object detection, coupled with an omnidirectional vehicle for interception of a free-falling object. This approach to object capture appears to be unprecedented in existing literature.

The team harnessed the full potential of available hardware, capturing a free-falling stress ball with varying success. However, the implications of this research extend far beyond this specific application. A similar vehicle could find application in more intricate dynamic systems, ranging from defense systems to space debris capture and other autonomous processes. These systems could also serve as platforms for researching advanced control systems to enhance efficiency, all powered by the capabilities of AI.

Furthermore, the combination of AI-powered image classification and object detection can be harnessed in complex environments such as industrial plants. With slight modifications, the current vehicle could perform diagnostics and maintenance using classification and detection, showcasing the potential for innovation in industrial processes.

In the pages that follow, a comprehensive account of the research conducted is presented, detailing the methodologies employed, the challenges encountered, and the outcomes achieved. This report serves as a testament to the boundless potential of AI-driven technologies and their transformative impact on the future of engineering.

II. Engineering Design

As mentioned previously, an omnidirectional vehicle was used in the current application of the object capture system. Risking the admittance of key information by oversimplification, an overview of the current system can be described as follows: A camera and computer onboard the vehicle detected the ball through AI while a control algorithm on the same onboard computer controlled the movement of the vehicle towards the object's landing position. The first and most challenging aspect of the task was to detect the ball.

A. Object Detection

The main reason the Jetson Nano was chosen was for its AI and deep learning capabilities, particularly its real-time inferencing abilities. These are powered by accelerated software, with all computing taking place on the edge. This project used a pre-trained object detection model that was initially trained on a broad dataset and fine-tuned to our needs through a method known as transfer learning. This learning process necessitated a custom dataset, requiring the team to gather thousands of images similar to those the onboard camera would see.

After taking several thousand images, they were uploaded to Roboflow, where the images were annotated, that is, manually inputted bounding boxes around the object in each image. Following this, a Convolutional Neural Network (CNN) was trained on the new dataset using a method called supervised learning. This training enabled the CNN, which works similar to a human brain, to establish a relationship with both the class of the object and its coordinates within the frame.

As a result, our system became fine-tuned to our specific case and was now able to make reasonably accurate predictions about the object's position on never-before-seen data, such as a live feed. After fine-tuning, the model was exported for inferencing. This export process formats the model so that it can be used in hardware-specific environments. Finally, with the converted model, we were able to carry out the inference process on the Jetson Nano. Fig. 1 presents a high-level view of the object detection system.

Ultimately, catching a thrown ball that has one second of airtime requires real-time inferencing capabilities. The Jetson Nano accomplishes its real-time inferencing by leveraging TensorRT, a machine learning framework from NVIDIA. This framework optimizes neural networks for low latency and

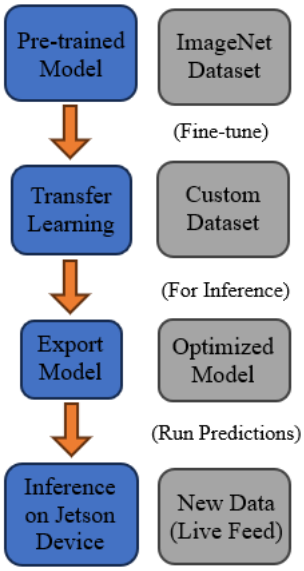

Fig. 1. The model takes multiple hours to train itself on the custom data set, where it makes connections similar to those made by a human brain.

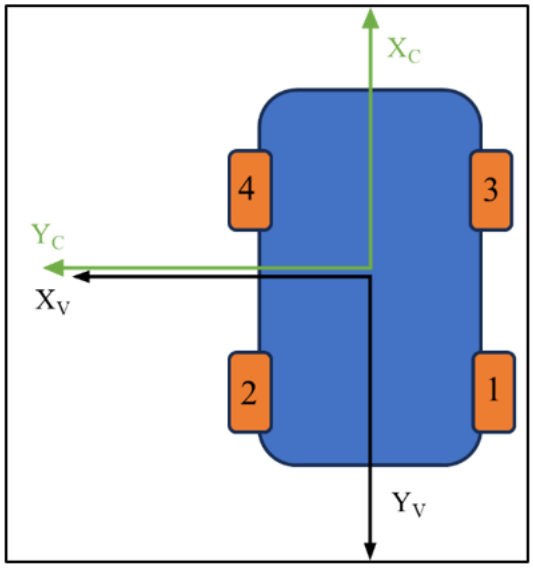

Fig. 2. Each wheel was numbered in accordance to the direction of the vehicle's reference frame, and was used to correctly implement the control algorithm.

high throughput using NVIDIA-specific GPUs, allowing for a performance boost up to 36X faster than CPU-only platforms and 6X faster than the PyTorch and TensorFlow frameworks during inference [2]. Once real-time inference is achieved, useful information about the object, such as class, confidence level, bounding box location, and dimensions, can all be extracted and used to control the vehicle.

B. Object Capture

With object detection established, the focus of this paper can now move to the control of the vehicle. The goal of the control algorithm implemented on the Jetson Nano was to position the vehicle such that the net attached to it would be directly under the object being detected. This way, the object would fall into the net, resulting in successful capture. Providing it makes a detection, the object detection system returns the bounding box of the detection. This, and its properties are currently the only inputs into the control algorithm. The first step in creating the control algorithm was to relate the coordinate system directions used to control the vehicle with the one created by the camera. Fig. 2 portrays the camera's coordinate system as well as the vehicle's, where C subscript denotes the camera and the V subscript denotes the vehicle.

The error vector is initially defined within the camera's frame and is then converted to a measurable unit rather crudely. To convert from pixels to a physical distance (mm), the conversion is found using the diameter of the ball and the width or height of the bounding box. This method would not work with objects of unknown size and shape. Additionally, the error value in the vehicle's y-direction was subtracted by the distance from the camera's center to the net's center to ensure the control would place the vehicle such that the object was above the net.

To convert the error values to engine inputs, two separate PD controllers were integrated. One controlled the lateral, positional movements of the vehicle, while the other controlled the angular displacement. This was chosen because the vehicle rotates easier than it translates, thus the coefficients would be largely different and need to be separately controlled. From the mathematical model of the vehicle, the necessary rotation for each wheel in radians per second can be described by the following equations:

$$\Psi_1 = \frac{1}{R}(V_y + V_x - (L+H)\omega) \tag{1}$$

$$\Psi_2 = \frac{1}{R}(V_y - V_x + (L+H)\omega) \tag{2}$$

$$\Psi_3 = \frac{1}{R}(V_y - V_x - (L+H)\omega) \tag{3}$$

$$\Psi_4 = \frac{1}{R}(V_y + V_x + (L+H)\omega) \tag{4}$$

Using these equations and normalizing the output, the motors were controlled to position the vehicle correctly.

Working in tandem, the object detection and control algorithm were implemented onto the Jetson Nano, using an Adafruit FeatherWing as the motor driver for the 4 DC motors. A battery housing and net attachment were 3D printed in order to make the system streamlined and effective. The omnidirectional vehicle then was able to detect the ball and move according to the inputs created by the control algorithm.

III. Research Methodology

With the conceptual and technical framework of the object capture system in place, our focus shifts to two key aspects: assembling the essential custom dataset for the CNN and assessing the vehicle's performance.

A. Data Collection

The success of the project relied not only on the proper implementation of AI and control algorithms but also on the rigorous gathering and handling of data. The data collection process was vital in training the system to detect and respond to the object effectively. The following will outline the data collection techniques utilized in the development and fine-tuning of the omnidirectional vehicle.

The data collected for our simplified object capture task were aimed at teaching the object detection algorithm, particularly the CNN, to recognize the appearance of the target object and to determine the appropriate size of the bounding boxes as the object moved through space. The size of the bounding boxes was of particular importance because it was used in our control algorithm as a conversion factor.

To achieve the desired level of accuracy, the camera was positioned to resemble what the omnidirectional vehicle would see, and videos of the ball being thrown in the air were recorded. The videos were then separated frame by frame, preparing them for annotation. Given that a rolling shutter camera was used, the data had to be meticulously combed through, and certain frames were deleted to avoid incorrect sizing of the bounding boxes. This deletion was vital, as objects moving at relatively high velocity could appear blurry and take up more pixels within the frame. If the dimensions of the bounding boxes were incorrect, the conversion factor would also be incorrect, which could introduce noisy data and send improper output signals to the motors.

Given our relatively simple object capture task, which involved capturing a specific ball against a plain background, we were able to optimize the detection algorithm for precision and speed. However, it is important to note that this optimization came at the expense of robustness, specifically with respect to other objects and backgrounds. To increase robustness for our particular case, we utilized Roboflow to create data augmentation files [6]. These data augmentation files, constructed from our existing data, introduced rotation to the existing images, providing more reference points that the CNN could use for prediction. This data collection technique also increased our dataset to a more appropriate size of 10,000 images for training.

To test the object detection algorithm, a new set of images was recorded and ran through the object detection algorithm from which we could gauge the success of the object detection.

B. Testing

Additionally, once we had implemented the full engineering design, we aimed to test the object capture system. For this purpose, we devised two different tests to capture data.

First, we performed a restricted test. We positioned the ball within 1 meter of the vehicle, allowing the vehicle to reposition itself under the ball before it was dropped. This test was conducted to see if the vehicle was able to adequately position itself to catch the ball.

Second, we performed a full test. We threw the ball in the air and observed how the omnidirectional vehicle autonomously positioned itself according to the control system's output, responding to the moving ball. This data allowed us to evaluate whether the system was working properly for the set goal, and ultimately, to determine the success of the project.

IV. Data Analysis

Having detailed our methodology and the specific approaches taken during data collection, we now shift our focus to the analysis and evaluation of the results obtained.

A. Object Detection Results

As mentioned previously, additional images were taken to test the object detection algorithm. From the test we were able to find the precision, recall, and F1-Score for different cases. Precision is the ratio of correctly predicted positive observations to the total predicted positives, recall is the ratio of correctly predicted positive observations to the total available positive predictions, and the F1-Score is the weighted average of Precision and Recall, which is used to calculate accuracy. They are calculated in the following ways. Where, True Positives (TP) are the data points in which the model correctly identified the object, False Positives (FP) are when the model incorrectly identified something as the object, and False Negatives (FN) are when the model incorrectly identified that the object was not present.

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

$$F1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{7}$$

After performing a test on 1,910 new images and adjusting the detection confidence threshold level to 50%, 40%, 30%, and 20%, we calculated the Precision, Recall, and F1-Score.

TABLE I
Object Detection Testing at Different Confidence Thresholds

Object Detection Tests			
Confidence Threshold(%)	***Precision***	***Recall***	***F1 Score***
50	0.99787	0.49162	0.65872
40	0.99007	0.62670	0.76755
30	0.97228	0.77120	0.86015
20	0.95349	0.90157	0.92680

The lowest confidence threshold had the highest F1-Score.

As demonstrated by the data in Table 1, when testing at a 50% confidence threshold (meaning any detection predicted below a 50% threshold was discarded), the Precision of

the model was almost 100%. However, both the Recall and F1-Score were significantly lower, leading to many missed detections. Given that the ball is in the air for an average time of one second and considering the object detection algorithm's performance of 40 frames per second, it becomes clear that not only high precision but also high accuracy was needed. This combination ensures that we detect the ball effectively within the limited time window. To address this issue, the confidence threshold parameter was lowered, leading to a slight decrease in Precision but a massive gain in both Recall and F1-Score, with all these metrics now above 90%.

We then proceeded with the analysis of test data for the object capture system. For the restricted test (as seen in Table 2), 100% of the balls were perfectly caught and none were completely missed. Considering the data, we can determine that the vehicle's object detection and control algorithms are working well together, and the vehicle is reaching the appropriate location that we determined for the ball within the frame.

For the full test, the ball was thrown with a trajectory following a narrow parabolic path, allowing for a realistic test of the vehicle's performance. Out of 100 throws, the data (Table 3) was not as convincing as the restricted test. The successful capture rate was 48%.

The results of our testing showcase both the strengths and areas of improvement in the vehicle's object detection and capture system. While the high precision and accuracy in controlled settings underline the system's potential, the need for variations in confidence thresholds and realistic throwing tests reveal the complexities involved in optimizing both accuracy of detection and real-time response.

Although the performance was satisfactory, it was heavily influenced by budget constraints, leading to hardware limitations such as the use of a smaller CNN and a PD control without future predictions. These constraints profoundly affected the vehicle's capabilities, impacting factors such as accuracy, reaction time, and resource utilization. These constraints and limitations of our current system will be explained further, while examining the potential future development of this project.

TABLE II
RESTRICTED TEST OF OBJECT CAPTURE SYSTEM

Object Capture Tests	
Result	***Percentage (%)***
Caught in Air	100
Caught After Bounce	0
Rim Contact	0
Miss	0

The restricted test outcomes were perfect.

V. DISCUSSION

One of the primary constraints that shaped the performance of our object capture system was the selection of hardware. For our project, we decided to use NVIDIA's Jetson Nano, an affordable yet powerful machine designed for machine learning and deployment at the edge. Despite its impressive handling and deployment of neural networks, the Jetson Nano's limited RAM memory of only 4 GB posed challenges, particularly given that neural networks are memory-intensive procedures. This constraint restricted our ability to implement more intricate solutions for object capture, such as utilizing larger, state-of-the-art object detection models like YOLO-v8, adding a Kalman filter, or integrating additional neural networks for object tracking and depth estimation to make accurate future predictions. Such an object capture pipeline could be implemented with newer, more advanced hardware such as the Jetson Orin series.

TABLE III
FULL TEST OF OBJECT CAPTURE SYSTEM

Object Capture Tests	
Result	***Percentage (%)***
Caught in Air	31
Caught After Bounce	17
Rim Contact	33
Miss	19

The full test outcomes were less promising.

The main limitations of our current hardware stem from the accuracy of the object detection algorithm in more complex testing environments, and the system's inability to predict the future position of the object. Currently, vibrations caused by the movement of the vehicle, coupled with the inescapable false positives stemming from the lowered confidence threshold level, lead to choppy detection and reduced accuracy. This situation results in incredibly noisy data being input into the system, affecting not only the final application but also the tuning of the PD control. As the team attempted to fine-tune the two PD controllers that govern the system's movement, it was challenging to determine whether small inconsistencies were due to chattering from detection or issues within the control system itself.

Additionally, the system's inability to predict future positions means that the system is always reacting to information, rather than calculating the ball's trajectory to intersect at the most probable location. While we attempted to overcome these limitations by adding a Kalman filter, the hardware's memory constraints delayed all processes, causing the system to be too slow to react in a real-time scenario, and making this approach unusable. Another avenue currently being explored for future prediction is the integration of an Inertial Measurement Unit (IMU) attached to the vehicle. This avenue is promising due to the ability to track the position of the vehicle and implementation of an extended space-time control.

Despite these hardware constraints and limitations, the system performed better than expected, demonstrating that our team's proof of concept successfully implemented a low-cost AI-powered object detection and capture system. This accomplishment suggests that the current system could be adapted to support other specific functions, such as image classification to identify the type of object or image the robot is viewing, as well as depth estimation for depth sensing scenarios. Adjusting

our current system to one of these capabilities would require additional training but could be implemented rather easily.

Additionally, mathematically modeling an omnidirectional vehicle can be extremely difficult due to the high amounts of slip experienced by each wheel, making a simulation of the system hard to obtain. An improvement that could qualm the object detection issues would not only help increase the accuracy of detections, but also performance of the control algorithm, creating a more accurate and rapid system. This would result in more successful captures and would only need to be facilitated by better hardware.

VI. Recommendations for Future Work

To combat the aforementioned deficiencies, there are multiple options. The first, rather obviously, would be to implement better hardware. Implementing enhanced hardware, including CPUs and GPUs with updated architecture and increased RAM, to allow for parallel execution of multiple AI and control algorithms. This would further expand the robotic capabilities. Secondly, implementation of Inertial Measurement Units (IMUs) to provide precise information about the orientation, acceleration, and rotational rate of the robot in three-dimensional space. Lastly, another factor that would help object detection specifically: incorporating a global shutter stereo camera for more accurate 3D imaging and depth information. The global shutter stereo camera would take better images of the ball as well as provide another dimension for control through depth estimation.

VII. Acknowledgement

The authors would like to thank Dustin Franklin for his valuable insights and assistance in understanding the concepts related to the training and deployment of object detection algorithms. His open-source materials, available on GitHub, were instrumental in the development of this research.

References

[1] Addison Sears-Collins. "Omni-Directional Wheeled Robot Simulation in Python." Available at: https://automaticaddison.com/ Created on: May 14, 2020.

[2] NVIDIA TensorRT. Available at: https://developer.nvidia.com/tensorrt?ncid=so-yout-997504#cid=dl13_so-yout_en-us Accessed on: August 7, 2023.

[3] Lady Ada. "Adafruit Stepper + DC Motor FeatherWing." Available at: https://learn.adafruit.com Updated on: July 29, 2016.

[4] Dustin Franklin. NVIDIA Jetson Developer. GitHub Repository. Available at: https://github.com/dusty-nv Accessed: July 10, 2023.

[5] NVIDIA Developer. "Jetson AI Fundamentals." YouTube Playlist. Last Updated: March 15, 2023. Available at: https://www.youtube.com/playlist?list=PL5B692fm6--uQRRDTPsJDp4o0xbzkoyf8

[6] Roboflow. "Roboflow Train." Website. Accessed on: July 28, 2023. Available at: https://roboflow.com/train