Iowa State University

Department of Electrical and Computer Engineering

Adaptive Visual Target Identification and Tracking through Convolutional Neural Network (CNN) based Machine Learning and Computer Vision

Alexis J. Renderos, William Lavelle, Rolf Anderson, Ritvik Maripally

Abstract

The aim of this project is to identify a chosen object of interest at range using a camera feed and an image processing algorithm to assist in the detection, identification and tracking of the target.

This solution was then implemented as a rail-mounted tactical scope attachment for a Nerf Blaster. The attachment provides the user with real-time feedback on their aim using a side mounted screen.

The Team

- Alexis J. Renderos – Project Lead, Chief Engineer
- William Lavelle – Operations Director
- Ritvik Maripally – CAD Design and Data Collection
- Rolf Anderson – Software Development

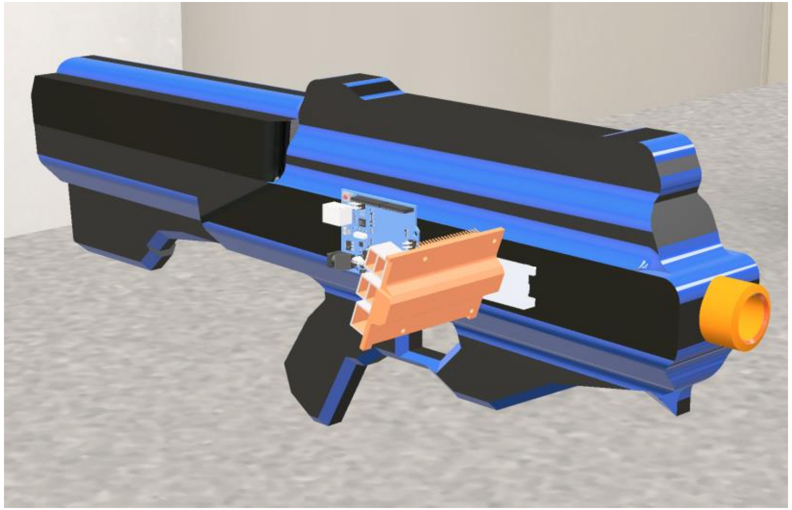
Collaboration and Communication

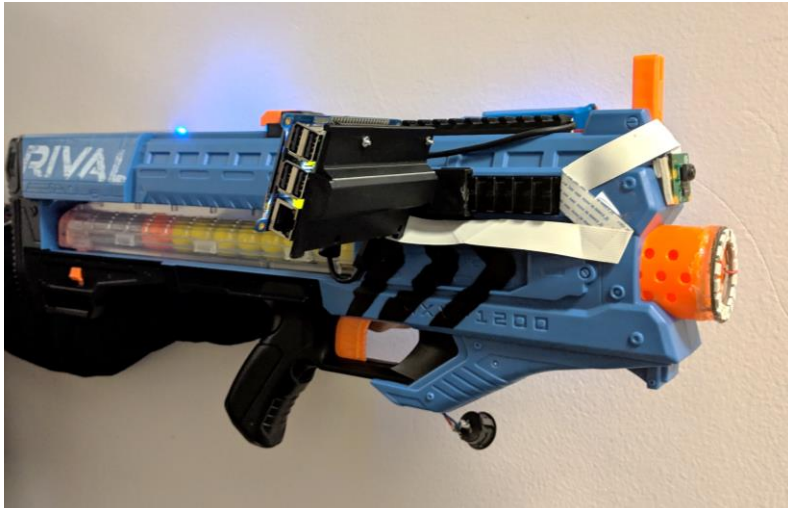
- GroupMe
  - We used this as the primary contact method between team members.
- GitHub Organization
  - We had two main repositories, one for code, and one for CAD. This allowed us to collaborate on different parts of the project easily and track contributions.
- Google Drive
  - Everything else that didn’t end up on one of the GitHub repositories ended up on a shared Google Drive. This was where most of our research and data went.
- Email
  - We used email as a more formal method of communicating with those outside of our project group, for example, Critical Tinkers Officers and our TA’s.

The Targets

- 3 Popular Drink Cans
  - Arizona Green Tea
  - Razzleberry Peace Tea
  - Monster Mango Loco Juice
- Distinct colors
- Easy to find in stores
- They taste great, too!

The Design





The Design

The Design

- Nerf Rival Zeus
  - The blaster!
- Raspberry Pi
  - Processes image data and displays feedback on aim
- Arduino
  - Controls the NeoPixel light ring
- NeoPixel Light Ring
  - Provides additional light to the subject if needed

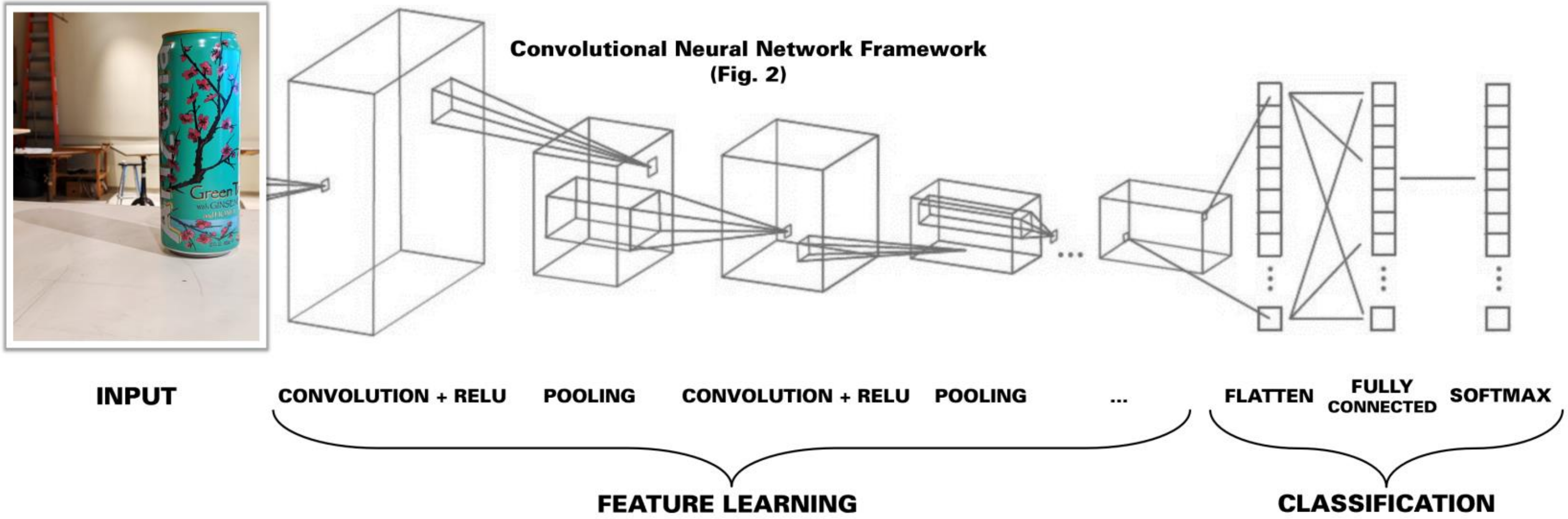
Hardware Features

- 40-50 Foot Effective Range
- Light Ring
  - Provides extra light to user and camera in low light conditions
- Side mounted LCD
  - Used to display camera feed and algorithm output to user
- Fully portable
  - Powered through a network of batteries mounted on the blaster
- Sounds like an angry lawnmower

Software

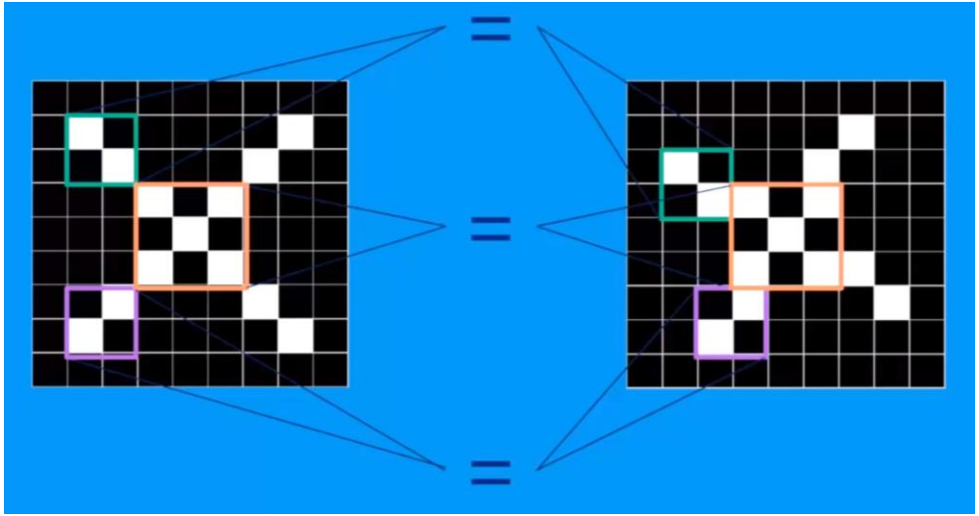
- Written all in Python
- OpenCV
  - A software library dedicated to real-time computer vision
- Tensorflow
  - A symbolic math library that is also used for machine learning applications
- Keras
  - An open-source neural-network library that runs on top of Tensorflow
- Plus a *little* C for the Arduino

Convolutional Neural Networks



Convolutional Neural Networks

- Also known as ConvNets
- Look for patterns in images



Convolutional Neural Networks

- Filtering
  - Takes image data and tries to find a feature it knows
- Built on what are called Convolutional Layers
  - Convolution
    - Repeated filtering across the entire image, produces a new "filtered image" that indicates to us where the feature lies on the image.
  - Pooling
    - Create a new, smaller image that further indicates the prevalence of a pattern.
  - ReLU (Rectified Linear Units)
    - Take any negative values and make them zero.

Convolutional Neural Networks

- Fully Connected Layer
  - Now the ConvNet tries to connect the dots and determine what it is looking at.
  - It will use the “images” produced by the convolution layers to vote on what the current image is.
  - This layer dynamically changes through gradient descent, which is essentially the changing the voting weight based on the ConvNet’s error.
- These layers all get stacked on top of each other
  - Like, *really, really* stacked

Convolutional Neural Networks

- In the end, the ConvNet should be able to:
  - Look through an image
  - Find any targets
  - Approximate their position
  - Analyze which target is in the field of view

KCF Tracking Algorithm

- Tracks a set of pixels
- Analyzes how it changes over time
    - Determines if that change is too drastic (tracking failure) or if that change is acceptable (tracking success).
- Computationally Efficient – Fast runtime
- Surprisingly accurate
- Handles target occlusion well

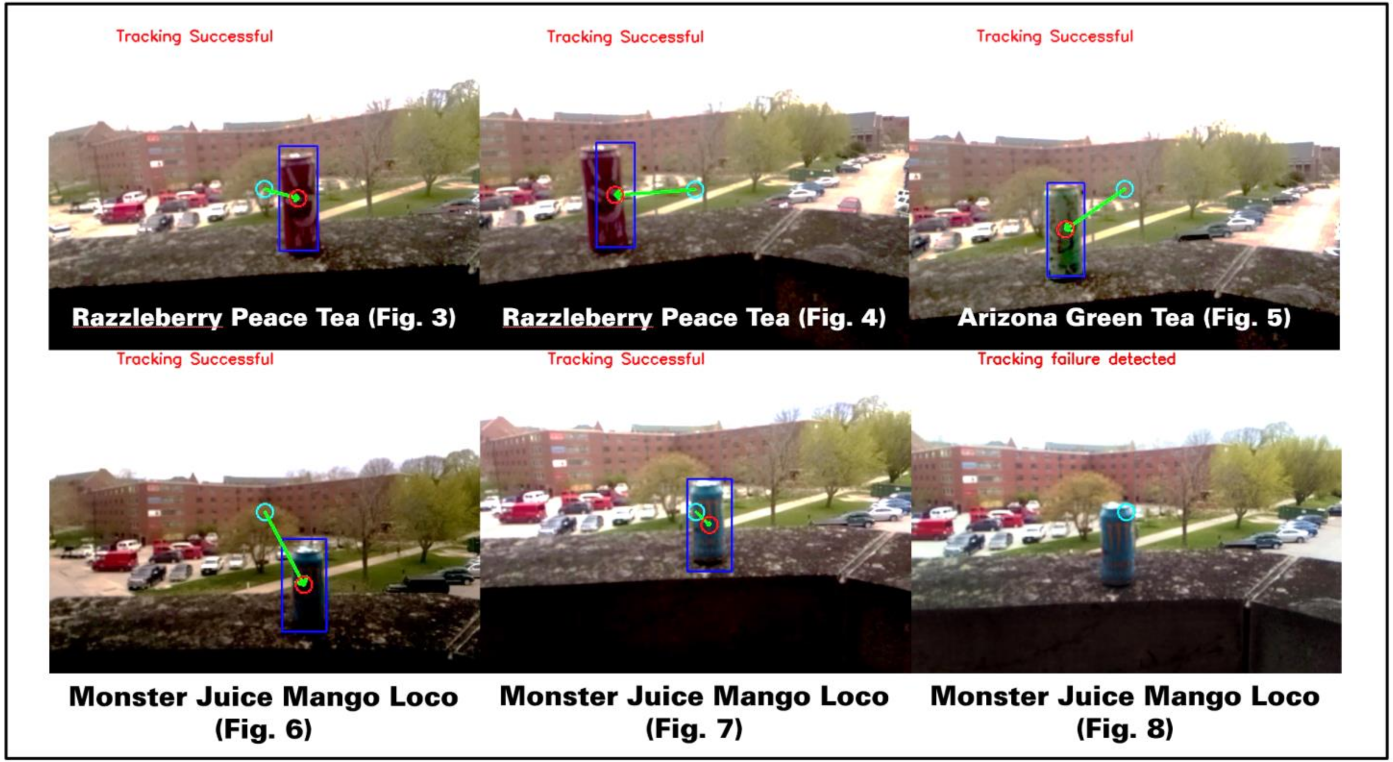
Software Flow

- ConvNet provides the initial location of the target to the KCF Tracker once a target is found
- KCF Tracker continues to find the location of the target until it fails
- Once the KCF Tracker fails, the ConvNet takes over again to repeat the process.

User Feedback

- Mounted on the Nerf Blaster is a screen that displays the current camera feed to the user.
- Gun Reticule: Light Blue Circle
- Target Center: Red Circle
- Target Location: Dark Blue Rectangle
- Error Feedback: Green Arrow

User Feedback



Conclusion and Analysis

- The CNN sucked.
  - Overall, target was too low of a resolution to be meaningful to the CNN, and thus performance suffered
  - CNN was trained twice, once with high-resolution target images, and once with low-resolution target images
  - The high-resolution model excelled with high-resolution targets (as expected)
  - Low-resolution model did well, but was easily thrown off by false positives
- The KCF Tracker slaps.
  - Fast, accurate, and recovered well from target occlusion.
  - 10/10

Conclusion and Analysis - Potential Applications

- The CNN relies on three major circumstances:
  - One, the target is visible (little to no occlusion)
  - Two, the target is consistent (all targets look the same and are predictable)
  - Three, the target has been previously analyzed (thousands of photos)
- With these in mind:
  - The potential military applications are slim—targets are not always consistent and have rarely been analyzed as much as the ConvNet demands.
  - A major retailer could use this to assist in a shipping warehouse—it could help workers/robots find the correct product.
  - This sort of technology is actually already in use in the Amazon app—you can scan a product (not necessarily the barcode) and it will show you listings for it.