NeuralKart: A Real-Time Mario Kart 64 AI

Harison Ho
Varun Ramesh
Eduardo Toress Montano
2017 - cs231n.stanford.edu

Abstract

a real-time Mario Kart 64 autopilot

two main components

-an omniscient search AI simulates different possible actions and generates a training set

-a convolutional neural network (CNN) trains

Introduction

Mario Kart 64

only looking at image of the screen

constantly hold accelerate

return a steering value

learn feature extraction with CNNs

Background / Related Work

Imitation Learning

Reinforcement Learning

Game Playing

Autonomous Vehicles

Mario Kart 64

Method

Bizhawk

emulater

-save/load states

-play for any number of frames

-access in-game memory locations

-save screenshots

Method

Search AI

determine the best steering action

-save current position as root state

-11 different steering values, simulates the results of the gameplay for 30 frames

reward

-a weighted sum of the current progress

-current kart speed

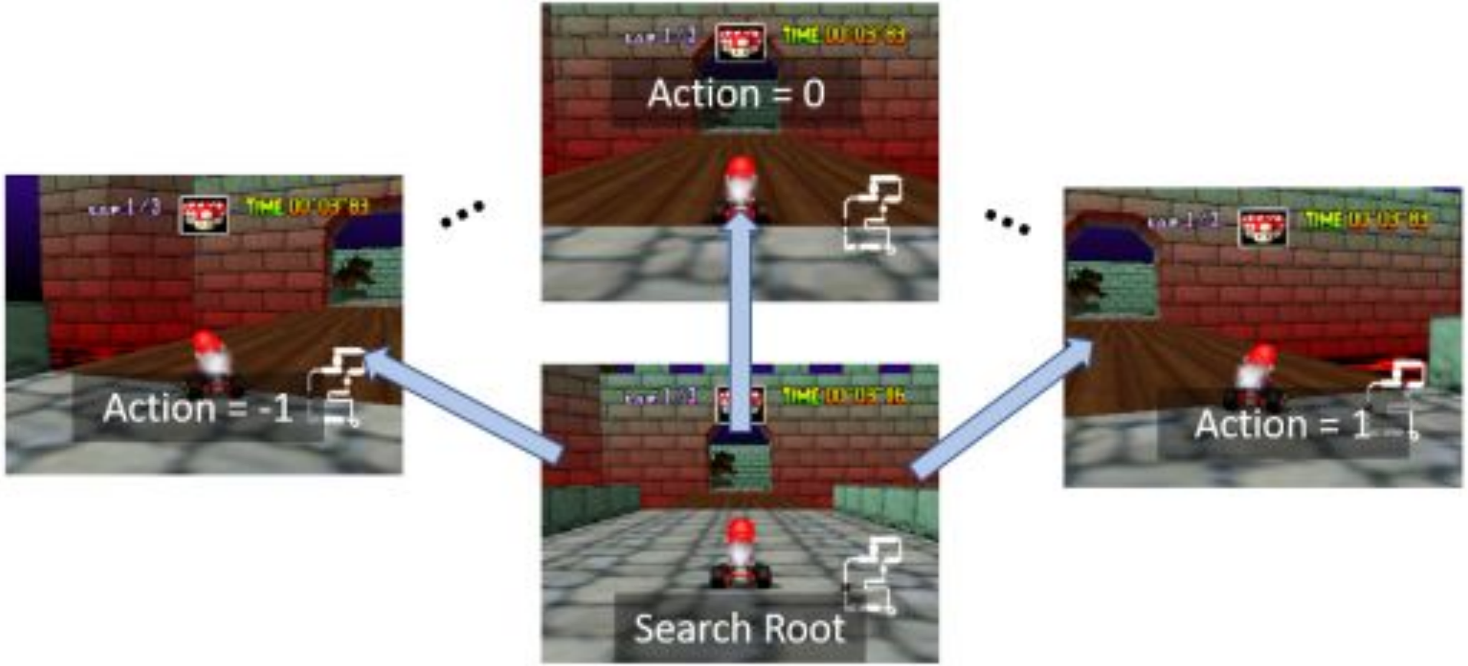

Figure 1. A demonstration of the search process. The search AI simulates the outcomes of 11 different angles, chooses the angle yielding the greatest progress, and stores the search root image and steering angle as a single datapoint.

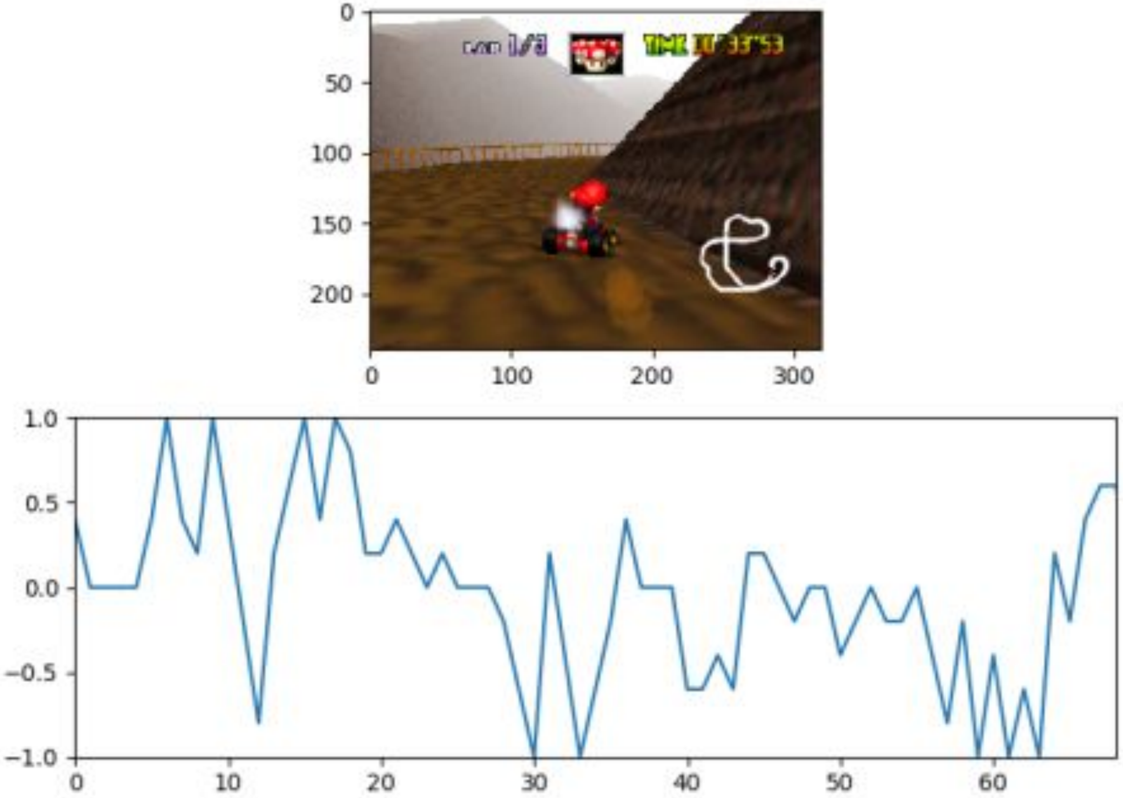

Figure 2. The bottom graph displays the steering values that the search AI has chosen up to the current point in time.

Method

Real-time CNN

-incorporates 5 batch

-normalization-2D

-convolution-ReLU layers

-5 dense layers

Method

DAGGER(Dataset Aggregation) Algorithm

-run the search AI

-initialize the weights of the CNN

-playing with predicted steering angles

-randomly pause and run the search AI from current point

-run the search AI for 120 frames and save image-steering angle pairs

-retrain the CNN with new data set

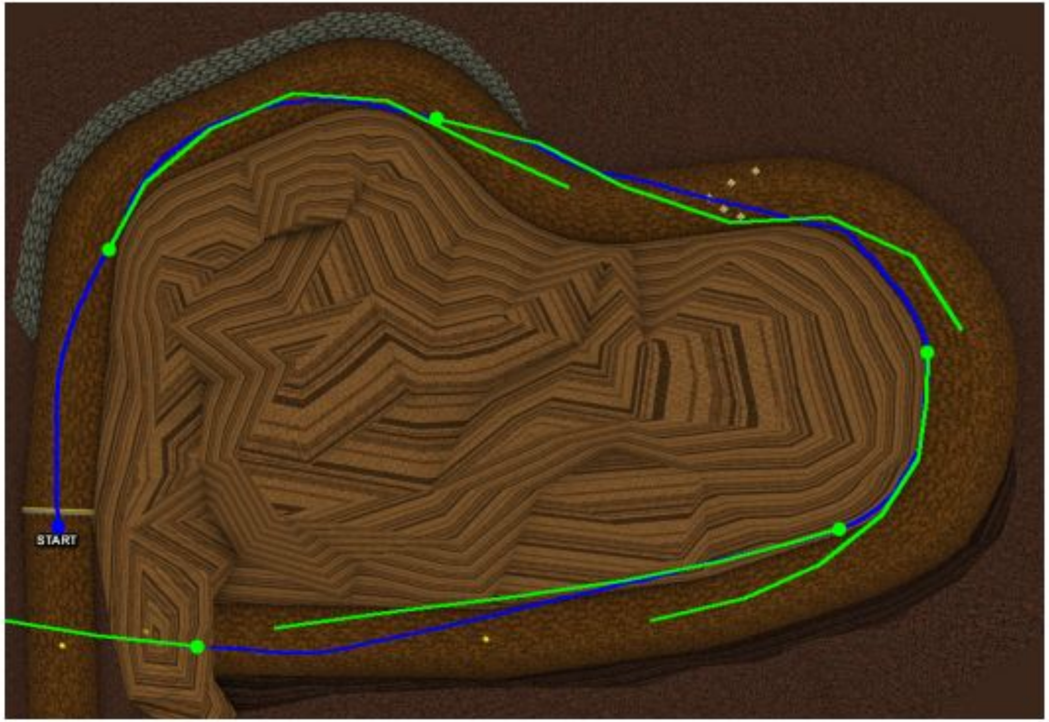

Figure 3. The paths that the kart takes using the DAGGER approach. The blue line shows the path of the CNN AI playing in real-time. The green lines show the trajectories chosen by the search AI when started at states randomly sampled from the CNN AI's play-through.

Method

Playing in Real-time

-takes a screenshot

-sends a request to the server

-receives the prediction

-sets the joystick value.

Method

Input Remapping

linearly interpolated our potential angles from range from -128 to 127

issues

-most of this space is a dead-zone

-the horizontal displacement of a turn is not linear w.r.t. the joystick value

solution

-a mapping function $J(s)$ that maps a “steer” input domain $s \in [-1, 1]$ to joystick values

- $\alpha(s) = (\mathrm{sgn}(s) \times \sqrt{(0.24 \times |s| + 0.01)} + 1) / 2$

- $J(s) = \mathrm{floor}(-128 \times (1 - \alpha(s)) + 127 \times \alpha(s))$

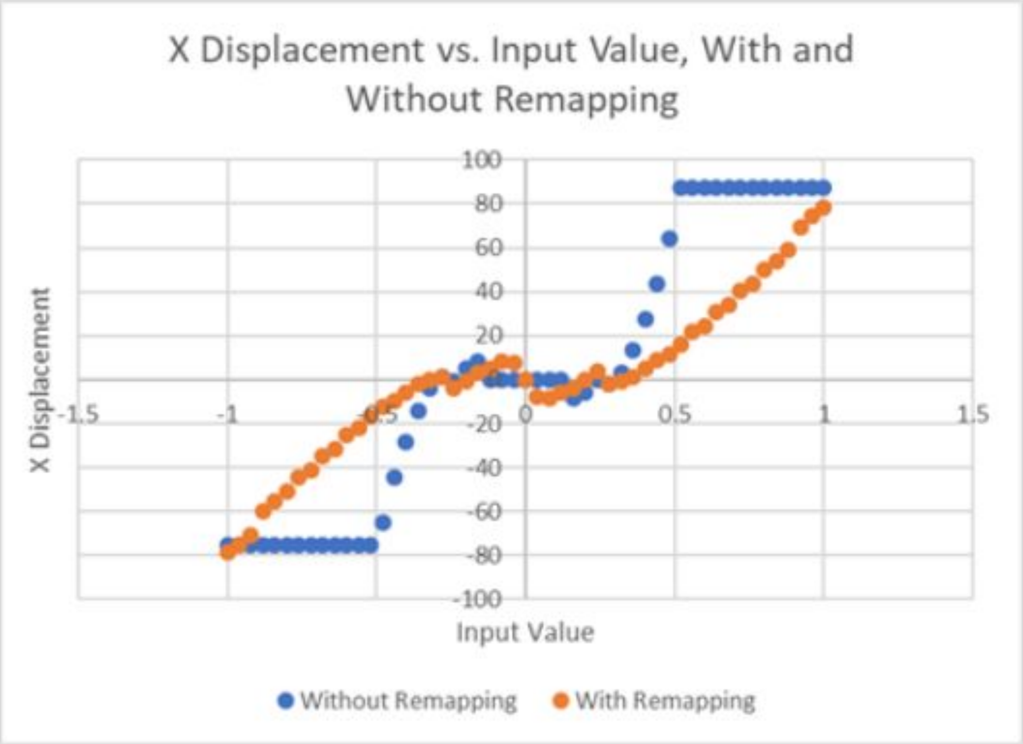

Figure 5. The horizontal displacement of the player with respect to the input trajectory is non-linear, and most of the values are taken up by dead zones. Our input remapping removes the dead-zones and makes the displacement linear with respect to the input value.

Results

Quantitative Evaluation

run 10 races in real-time and calculate the mean race time

Track	Autopilot Time (s)	Human Time (s)
Moo Moo Farm	97.46	94.07
Luigi's Raceway	129.09, 1 DNF*	125.30
Choco Mountain	138.37, 2 DNF*	129.50
Rainbow Road	389.18	365.60

Table 1. Achieved track times for the autopilot bot and the human; the autopilot times have been averaged over 10 runs. *DNF signifies that the autopilot got stuck and was unable to finish some number of races.

Results

Qualitative Evaluation

-stable to perturbations by an external force [figure7]

-The steering behavior resembles how a human would play Mario Kart

- AI is able to ignore the information added by new elements, despite never having seen those elements before [figure 8]

-situations where the autopilot would slow down

slide against walls or drive on the edge of the road next to sand or grass

Figure 7. On Luigi's Raceway, our AI is stable to perturbations. Here, an actual joystick is overriding our AI, pushing it to the right. However, the AI correctly sees that the proper response is to turn to the left. We don't observe the same level of stability on every track.

Figure 8. Our AI is trained in Time-Trial mode, but can still race in Grand Prix mode. Grand Prix introduces new UI elements, item boxes, opponents, and hazards like bananas.

Experiments

Image Reflection

-model may not be looking at the curvature of the road

Classification-based Model

-model performed worse than the regression model

Experiments

Training on All Tracks Together

Track	Individual Data (s)	All Data (s)
Moo Moo Farm	97.46	97.63
Luigi's Raceway	129.09, 1 DNF	129.03
Choco Mountain	138.37, 2 DNF	131.93, 3 DNF
Rainbow Road	389.18	396.74, 1 DNF

Table 2. The performance of our model when trained on all of the data at once, versus keeping a separate dataset and weights file for each track. DNF signifies that some runs did not finish.

Beam Search

-increased the flexibility of the search AI

-play more difficult map

Conclusion

end to end neural systems can yield good performance as real-time controllers in games like Mario Kart 64