CREATING A PLAYER AID FOR GEOMETRY DASH APS360 PROGRESS REPORT

Ian Lu
Student# 1009972139
i.lu@mail.utoronto.ca

Jaden Dai
Student# 1009972228
jaden.dai@mail.utoronto.ca

Joel Vadakken
Student# 10010089798
j.vadakken@mail.utoronto.ca

Skyler Han
Student# 1009794830
hs.han@mail.utoronto.ca

ABSTRACT

In this report, our group outlines the process used to create a model, and assess its performance, to identify obstacles in Robtop's platformer video game, Geometry Dash. By taking a screenshot of the game, our model creates a semantic segmentation map of 7 classes of elements present on the screen, including the player, spikes, and platforms. —-Total Pages: 8

1 INTRODUCTION

We created a model that could take a screenshot from Geometry Dash and detect the locations of various elements in the game. Geometry Dash is a one-button platformer video game created by Robtop where a player controls an icon to avoid obstacles and collisions as they progress through a map. The goal of our project was to create a model that can detect the location of elements such as these obstacles. For example, our model should be able to take a screenshot such as in Fig. **??** on the left, and generate the image on the right, identifying the collision boxes of the spikes, the collision boxes of the platform, the location of the orb, the location of the player, and the background.

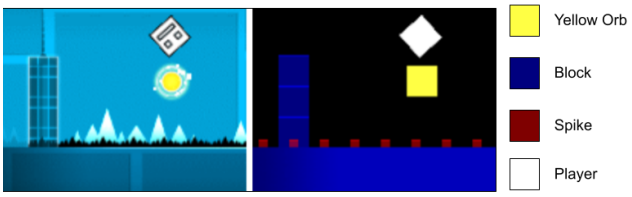

Figure 1: On the left is a sample screenshot that our model should be able to take in as input. The image on the right is an example of what the output of our model should look like, and the table contains the classes that the image was broken up into.Note that Geometry Dash collision boxes are strangely shaped for some elements such as the spike objects.

We believe that machine learning is an appropriate approach for this task due to the difficulties that can arise in identifying game elements when complex situations arise. For example, decorative objects can easily be recognized by a player as irrelevant but may trick an image-matching or hard-coded algorithm. Essentially, the obstacles in a game are always positioned in such a way that a player has a way forward, so part of being able to identify these game objects would be to understand what makes sense in the context of the game. An example is provided in Fig 2.

We believe this project is appropriate for deep learning due to the many complexities and edge cases that may appear. For instance, there are many decorative objects that a player can recognize as irrelevant but may trick a traditional hard-coded or image-matching algorithm.

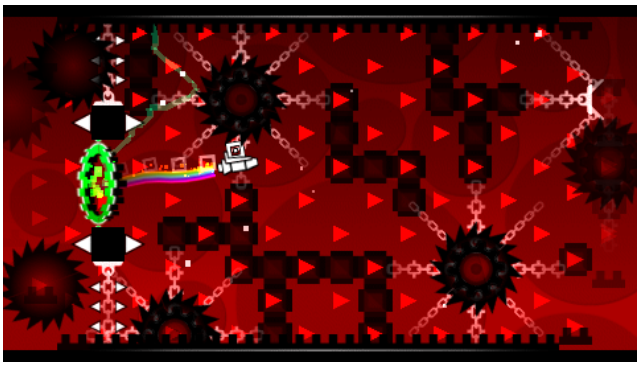

Figure 2: A training sample image from the level "Death Moon", this is an example of a screenshot traditional algorithms might have a hard time labelling. Note that the black saw blades may damage the player, while the red arrows, white chains, and black squares are decorative. However, to identify this, an algorithm would have to recognize that if the red triangles or black squares were not background objects, the player would have no way to proceed in this level.

This model can be a useful tool for players who are unable to access other paid tools to view collision boxes. Geometry Dash has a large community and player base, and many of the custom-made levels are very tough. Many players download mods to display hitboxes as they go through these levels so that they can improve their runs through practice. Our model can provide this service for free. Furthermore, this may be a useful model for future work in training an RL model to play Geometry Dash by simplifying inputted information.

2 Background and Related Works

Figure 3: A screenshot of CodeNoodle's visualization of his neural network from his Youtube video where he creates a bot to play Geometry Dash (CodeNoodles, 2022).

CodeNoodles (2022) created a simple Neural Network that could play a simplified version of Geometry Dash. His bot took in input information, such as the distance to the next block and the location of spikes, and outputted an action. See Fig 3. However, since CodeNoodles created a simplified version of Geometry Dash that could readily supply his bot with the needed information, his bot would not work on the actual game itself as the version of Geometry Dash that he made simplified the process of creating a bot immensely. For example, his bot did not have to worry about any image-processing. Our model, on the other hand, is solely concerned with this image processing, and as a result, there is exciting potential in connecting the output of our model to the input of CodeNoodles' bot to create a bot that can play on the official version of Geometry Dash.

Montalvo J. (2023) researched the efficacy of using semantic segmentation to preprocess the inputs to a RL model learning to play Super Mario. They found that this reduced the training time and improved the performance of their model. This report provides justification that creating these semantic segmentation maps of the screenshots from Geometry Dash can help train RL models in the future. See Fig 4

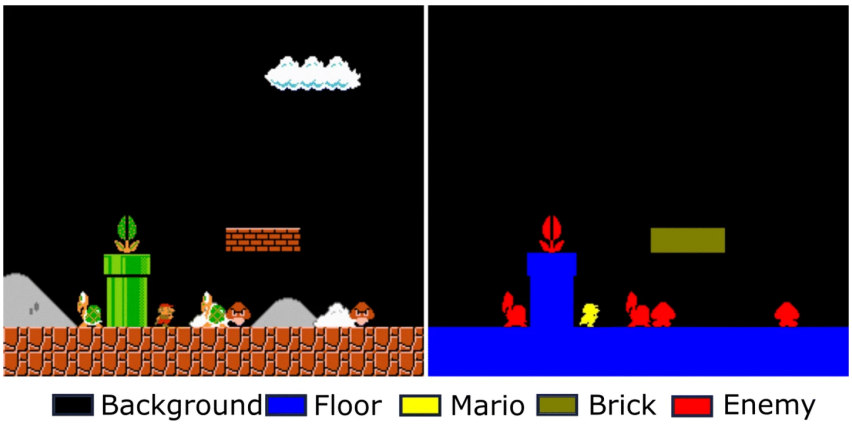

Figure 4: This picture is from an article that proved preprocessing images with semantic segmentation can help improve model performance, suggesting that our project has potential for use in a Geometry Dash bot in the future. (Montalvo J., 2023)

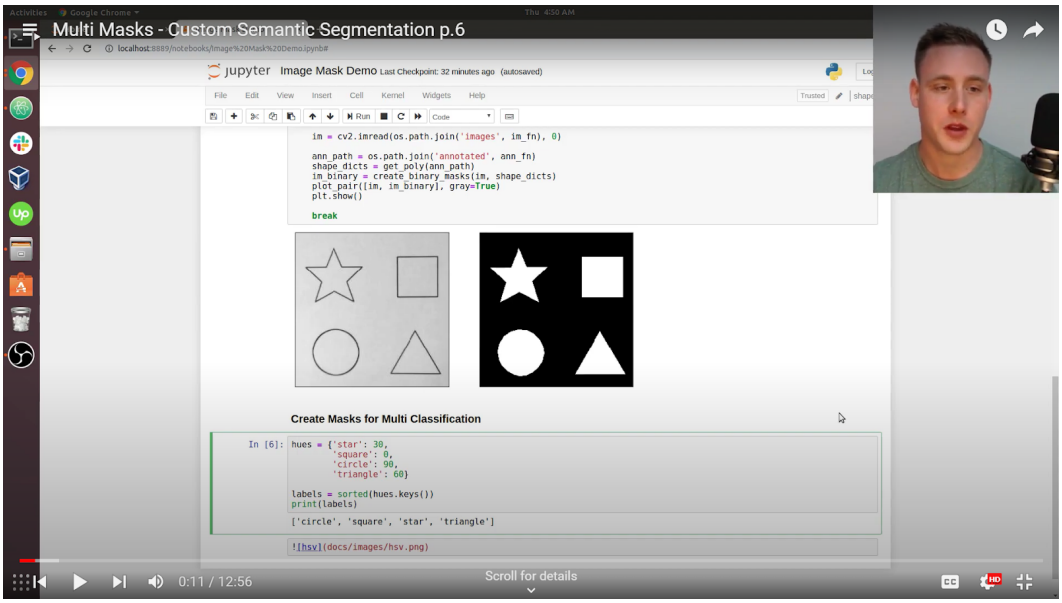

Figure 5: A screenshot of one of the video tutorials by Seth Adams outlining how to create a semantic segmentation map.(Adams, 2020)

Adams (2020) created a youtube tutorial on how to create semantic segmentation maps using tensorflow, and uploaded his code to GitHub. Since this was exactly what we needed for label creation and data-preprocessing, we carefully went through his work. Since our code needed to use pytorch as opposed to tenserflow, we had to heavily modify his code, but nonetheless, our work is built off of his. His code allowed us to create the labels and dataloader functions that we needed. A screenshot of one of his tutorials can be seen in Fig 5.

Lamba (2019) outlines an architecture that was used to build a model that could make semantic segmentations on biomedical images. It was called UNET and it solved a problem similar to ours. As a result, our model uses a similar architecture.

The SIMA Team (2024) at Deepmind created a model, called SIMA, that can take visual observations from a game to carry out an instruction by outputting keyboard events. Although their model does not achieve the same end goal as us (following instruction vs autonomous gameplay), their

agent's ability to understand complex visual information and map it to keyboard inputs supports our approach of utilizing screen capture for object detection and behavior planning.

2.1 RETRIEVAL OF STYLE FILES

The file `APS360_Project.pdf` contains these instructions and illustrates the various formatting requirements your APS360 paper must satisfy. Submissions must be made using LaTeX and the style files `iclr2022_conference.sty` and `iclr2022_conference.bst` (to be used with LaTeX2e). The file `APS360_Project.tex` may be used as a "shell" for writing your paper. All you have to do is replace the author, title, abstract, and text of the paper with your own.

The formatting instructions contained in these style files are summarized in sections 3, 4, and 5 below.

3 GENERAL FORMATTING INSTRUCTIONS

The text must be confined within a rectangle 5.5 inches (33 picas) wide and 9 inches (54 picas) long. The left margin is 1.5 inch (9 picas). Use 10 point type with a vertical spacing of 11 points. Times New Roman is the preferred typeface throughout. Paragraphs are separated by 1/2 line space, with no indentation.

Paper title is 17 point, in small caps and left-aligned. All pages should start at 1 inch (6 picas) from the top of the page.

Authors' names are set in boldface, and each name is placed above its corresponding address. The lead author's name is to be listed first, and the co-authors' names are set to follow. Authors sharing the same address can be on the same line.

Please pay special attention to the instructions in section 5 regarding figures, tables, acknowledgments, and references.

There will be a strict upper limit of 9 pages for the main text of the initial submission, with unlimited additional pages for citations.

4 HEADINGS: FIRST LEVEL

First level headings are in small caps, flush left and in point size 12. One line space before the first level heading and 1/2 line space after the first level heading.

4.1 HEADINGS: SECOND LEVEL

Second level headings are in small caps, flush left and in point size 10. One line space before the second level heading and 1/2 line space after the second level heading.

4.1.1 HEADINGS: THIRD LEVEL

Third level headings are in small caps, flush left and in point size 10. One line space before the third level heading and 1/2 line space after the third level heading.

5 CITATIONS, FIGURES, TABLES, REFERENCES

These instructions apply to everyone, regardless of the formatter being used.

5.1 CITATIONS WITHIN THE TEXT

Citations within the text should be based on the `natbib` package and include the authors' last names and year (with the "et al." construct for more than two authors). When the authors or the publication are included in the sentence, the citation should not be in parenthesis using `\citet{}` (as in "See Hinton et al. (2006) for more information."). Otherwise, the citation should be in parenthesis using

`\citep{}` (as in "Deep learning shows promise to make progress towards AI (Bengio & LeCun, 2007).").

The corresponding references are to be listed in alphabetical order of authors, in the REFERENCES section. As to the format of the references themselves, any style is acceptable as long as it is used consistently.

To cite a new paper, first, you need to add that paper's BibTeX information to `APS360_ref.bib` file and then you can use the `\citep{}` command to cite that in your main document.

5.2 FOOTNOTES

Indicate footnotes with a number[1] in the text. Place the footnotes at the bottom of the page on which they appear. Precede the footnote with a horizontal rule of 2 inches (12 picas).[2]

5.3 FIGURES

All artwork must be neat, clean, and legible. Lines should be dark enough for purposes of reproduction; art work should not be hand-drawn. The figure number and caption always appear after the figure. Place one line space before the figure caption, and one line space after the figure. The figure caption is lower case (except for first word and proper nouns); figures are numbered consecutively.

Make sure the figure caption does not get separated from the figure. Leave sufficient space to avoid splitting the figure and figure caption.

You may use color figures. However, it is best for the figure captions and the paper body to make sense if the paper is printed either in black/white or in color.

Figure 6: Sample figure caption. Image: ZDNet

5.4 TABLES

All tables must be centered, neat, clean and legible. Do not use hand-drawn tables. The table number and title always appear before the table. See Table 1.

Place one line space before the table title, one line space after the table title, and one line space after the table. The table title must be lower case (except for first word and proper nouns); tables are numbered consecutively.

[1] Sample of the first footnote
[2] Sample of the second footnote

Table 1: Sample table title

PART	DESCRIPTION
Dendrite	Input terminal
Axon	Output terminal
Soma	Cell body (contains cell nucleus)

6 DEFAULT NOTATION

In an attempt to encourage standardized notation, we have included the notation file from the textbook, *Deep Learning* Goodfellow et al. (2016) available at `https://github.com/goodfeli/dlbook_notation/`. Use of this style is not required and can be disabled by commenting out `math_commands.tex`.

Numbers and Arrays

a	A scalar (integer or real)
$\boldsymbol{a}$	A vector
$\boldsymbol{A}$	A matrix
$\mathsf{\mathbf{A}}$	A tensor
$\boldsymbol{I}_n$	Identity matrix with n rows and n columns
$\boldsymbol{I}$	Identity matrix with dimensionality implied by context
$\boldsymbol{e}^{(i)}$	Standard basis vector $[0, \dots, 0, 1, 0, \dots, 0]$ with a 1 at position i
$\text{diag}(\boldsymbol{a})$	A square, diagonal matrix with diagonal entries given by $\boldsymbol{a}$
a	A scalar random variable
$\mathbf{a}$	A vector-valued random variable
$\mathbf{A}$	A matrix-valued random variable

Sets and Graphs

$\mathbb{A}$	A set
$\mathbb{R}$	The set of real numbers
$\{0, 1\}$	The set containing 0 and 1
$\{0, 1, \dots, n\}$	The set of all integers between 0 and n
$[a, b]$	The real interval including a and b
$(a, b]$	The real interval excluding a but including b
$\mathbb{A} \backslash \mathbb{B}$	Set subtraction, i.e., the set containing the elements of $\mathbb{A}$ that are not in $\mathbb{B}$
$\mathcal{G}$	A graph
$Pa_{\mathcal{G}}(\mathrm{x}_i)$	The parents of x_i in $\mathcal{G}$

Indexing

a_i	Element i of vector $\boldsymbol{a}$, with indexing starting at 1
a_{-i}	All elements of vector $\boldsymbol{a}$ except for element i
$A_{i,j}$	Element i, j of matrix $\boldsymbol{A}$
$\boldsymbol{A}_{i,:}$	Row i of matrix $\boldsymbol{A}$
$\boldsymbol{A}_{:,i}$	Column i of matrix $\boldsymbol{A}$
$\mathsf{A}_{i,j,k}$	Element (i, j, k) of a 3-D tensor A
$\mathsf{A}_{:,:,i}$	2-D slice of a 3-D tensor
a_i	Element i of the random vector $\mathbf{a}$

Calculus

$\frac{dy}{dx}$	Derivative of y with respect to x
$\frac{\partial y}{\partial x}$	Partial derivative of y with respect to x
$\nabla_{\boldsymbol{x}} y$	Gradient of y with respect to $\boldsymbol{x}$
$\nabla_{\boldsymbol{X}} y$	Matrix derivatives of y with respect to $\boldsymbol{X}$
$\nabla_{\mathsf{X}} y$	Tensor containing derivatives of y with respect to X
$\frac{\partial f}{\partial \boldsymbol{x}}$	Jacobian matrix $\boldsymbol{J} \in \mathbb{R}^{m \times n}$ of $f : \mathbb{R}^n \to \mathbb{R}^m$
$\nabla^2_{\boldsymbol{x}} f(\boldsymbol{x})$ or $\boldsymbol{H}(f)(\boldsymbol{x})$	The Hessian matrix of f at input point $\boldsymbol{x}$
$\int f(\boldsymbol{x}) d\boldsymbol{x}$	Definite integral over the entire domain of $\boldsymbol{x}$
$\int_{\mathbb{S}} f(\boldsymbol{x}) d\boldsymbol{x}$	Definite integral with respect to $\boldsymbol{x}$ over the set $\mathbb{S}$

Probability and Information Theory

$P(\mathrm{a})$	A probability distribution over a discrete variable
$p(\mathrm{a})$	A probability distribution over a continuous variable, or over a variable whose type has not been specified
$\mathrm{a} \sim P$	Random variable a has distribution P
$\mathbb{E}_{\mathrm{x} \sim P}[f(x)]$ or $\mathbb{E} f(x)$	Expectation of $f(x)$ with respect to $P(\mathrm{x})$
$\mathrm{Var}(f(x))$	Variance of $f(x)$ under $P(\mathrm{x})$
$\mathrm{Cov}(f(x), g(x))$	Covariance of $f(x)$ and $g(x)$ under $P(\mathrm{x})$
$H(\mathrm{x})$	Shannon entropy of the random variable x
$D_{\mathrm{KL}}(P \Vert Q)$	Kullback-Leibler divergence of P and Q
$\mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$	Gaussian distribution over $\boldsymbol{x}$ with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$

Functions

$f : \mathbb{A} \to \mathbb{B}$	The function f with domain $\mathbb{A}$ and range $\mathbb{B}$
$f \circ g$	Composition of the functions f and g
$f(\boldsymbol{x}; \boldsymbol{\theta})$	A function of $\boldsymbol{x}$ parametrized by $\boldsymbol{\theta}$. (Sometimes we write $f(\boldsymbol{x})$ and omit the argument $\boldsymbol{\theta}$ to lighten notation)
$\log x$	Natural logarithm of x
$\sigma(x)$	Logistic sigmoid, $\dfrac{1}{1 + \exp(-x)}$
$\zeta(x)$	Softplus, $\log(1 + \exp(x))$
$\|\|\boldsymbol{x}\|\|_p$	L^p norm of $\boldsymbol{x}$
$\|\|\boldsymbol{x}\|\|$	L^2 norm of $\boldsymbol{x}$
x^+	Positive part of x, i.e., $\max(0, x)$
$\mathbf{1}_{\text{condition}}$	is 1 if the condition is true, 0 otherwise

7 FINAL INSTRUCTIONS

Do not change any aspects of the formatting parameters in the style files. In particular, do not modify the width or length of the rectangle the text should fit into, and do not change font sizes (except perhaps in the REFERENCES section; see below). Please note that pages should be numbered.

AUTHOR CONTRIBUTIONS

If you'd like to, you may include a section for author contributions as is done in many journals. This is optional and at the discretion of the authors.

ACKNOWLEDGMENTS

Use unnumbered third level headings for the acknowledgments. All acknowledgments, including those to funding agencies, go at the end of the paper.

REFERENCES

Seth Adams. Semantic-shape, 2020. URL https://github.com/seth814/Semantic-Shapes/tree/master.

Yoshua Bengio and Yann LeCun. Scaling learning algorithms towards AI. In *Large Scale Kernel Machines*. MIT Press, 2007.

CodeNoodles. Ai learns to play geometry dash —— final, Oct 2022. URL https://www.youtube.com/watch?v=A3JJsjZLRSok.

Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. *Deep learning*, volume 1. MIT Press, 2016.

Geoffrey E. Hinton, Simon Osindero, and Yee Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006.

Harshall Lamba. Understanding semantic segmentation with u-net. https://towardsdatascience.com/understanding-semantic-segmentation-with-unet-6be4f42d4b47, 2019. Accessed: 14 August 2024.

Bescós J. Montalvo J., García-Martín Á. Exploiting semantic segmentation to boost reinforcement learning in video game environments. In *Multimed Tools Appl 82*, 2023. doi: 10.1007/s11042-022-13695-1.

The SIMA Team. A generalist ai agent for 3d virtual environments. 2024. URL https://deepmind.google/discover/blog/sima-generalist-ai-agent-for-3d-virtual-environments/.