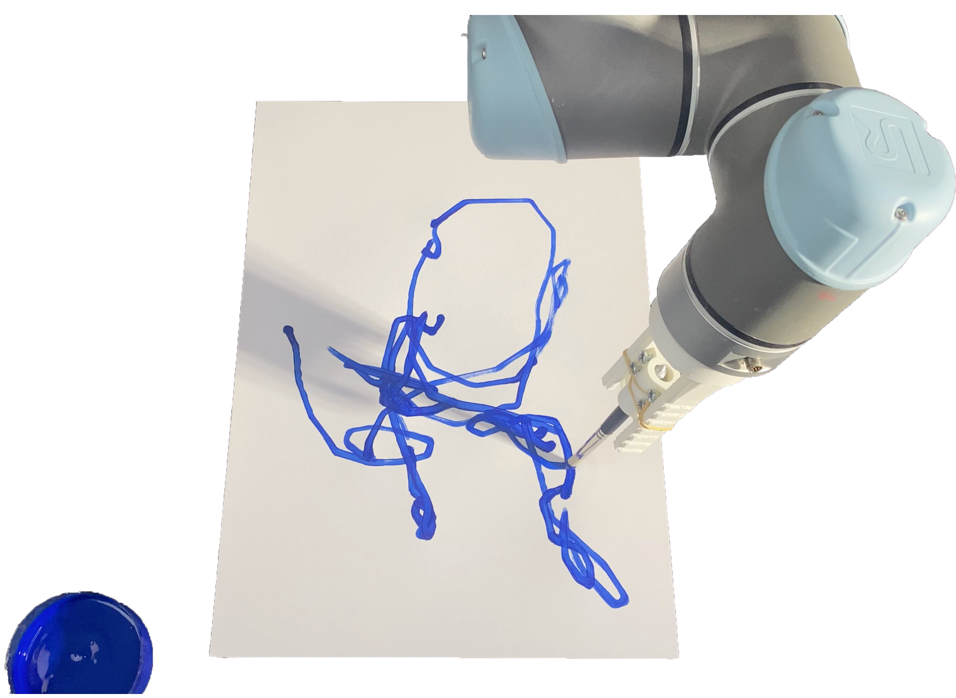
Algorithmic Muse: Robotic-Arm Learns to Draw Humans

Alessandro Mac-Nelly
Universitat der Kunste
MA Design & Computation
a.mac-nelly@udk-berlin.de

Abstract

This study explores training an AI Painting-Model, Algorithmic Muse, with Reinforcement Learning (RL) to draw abstract paintings of humans. Text to Image generators can already produce art images, but results are only pixel images representing the final result. Instead, our focus lies on how the drawing is created as a process of actions, including the brush strokes and calculating these strokes step by step. Picasso's Bull drawings 1945 are an example of this technique. Building a reward function that leads to good training results is a central topic this research focused on, and it includes evaluating artistic outcomes, evaluating a stroke, and giving the agent incentives to explore the canvas. This necessary training environment was built based on Gymnasiums Env Class and is compatible with standard RL frameworks. The Proximal Policy Optimization (PPO) implementation from Stable-Baselines3 is used to train in the drawing environment generating positive results. In practice, a mix of AI predictions and a tree search to compare potential strokes was used to create a drawing. The study including 203 subjects, evaluated and ranked 80 AI drawings, showing that some AI generated artwork is as aesthetically pleasing as human drawings. The vision of this project is to make Algorithmic Muse, while being connected to a robotic arm, collaborate with human artists in drawings where they take turns in the process.



I. Problem

Generative Adversarial Networks (GANs) [3] have significantly impacted the art scene, evidenced by Jason M. Allen's AI-generated artwork win at the 2022 Colorado State Fair[1]. These technologies, like Midjourney[2] and Dall-E2[3], create art from text prompts. Yet, they face criticism for lacking the nuanced, iterative creation process inherent in human artistry and for potential intellectual property issues, as they replicate the features of their training datasets [14].

To overcome this, I propose Algorithmic Muse, a novel AI Painting Model designed to mimic the nuanced human approach to drawing, where the focus lies on the creation process. Using an Agent trained with Reinforcement Learning in a drawing environment native to humans, either a digital environment or a robotic arm on a canvas, enables the Agent to assimilate the human drawing process.

The key problem this research tries to solve is how to simulate such an environment that closely resembles the human drawing process, not only in technique but also in the way creative decisions are made. The Agent has specific possibilities of moving and drawing on the canvas, so called actions. Having a too large action space can lead to difficulties for the Agent to learn a policy [6], but a too small one limits the drawing complexity. Furthermore, quantifying each action and movement a stroke is made up from means evaluating art in the process. This reward function must apply from the beginning to the final stage of a drawing.

I. State of the art

The journey of machines and algorithms in art creation dates back to the mid-20th century, exemplified by Jean Tinguely's "Machine à dessiner No. 3"[4], a precursor to today's GANs like Dalle-2 and Midjourney. The evolution into AI has significantly enhanced artistic possibilities, where the effectiveness of text prompts, including negative prompts, plays a crucial role in the outcome [8, 10]. Additionally, GANs ability to transform sketches or even brain waves into art marks a significant advancement [7, 11].

Looking at research considering Reinforcement Learning to draw humanly, the work of T. Zhou et al. [6] is interesting as they developed a model using DeepQ-Networks (DQN) that learned to scribble based on a given reference image. Similar to this problem, the agent faces a large action and observation space, making it hard for the model to converge in the training process. They were able to have positive results using DQN in combination with recorded stroke demonstrations of human artists. PaintBot [5] and a Model developed by Z. Huang et al. [4] can generate brush strokes to assimilate a given reference image even with colors. Instead, Algorithmic Muse is about creating artworks in a recursive manner where the observation of the canvas leads to the following action. With this method, the agent reacts to either its previous strokes or can collaborate with a human artist.

II. Training Environment

This custom environment is available on Github[5] and is compatible with the Gymnasium Env class[6]. The drawing mechanics work like this, for the Agent to make a stroke in this environment, every step and action creates a point next to the current point. This new point is generated with a vector length of 30 or 90 pixels in 12 radial directions. With the points' coordinates comes the decision of whether to

[1] https://www.nytimes.com/2022/09/02/technology/ai-artificial-intelligence-artists.html

[2] https://www.midjourney.com/home

[3] https://openai.com/dall-e-2

[4] https://www.tinguely.ch/en/tinguely-collection-coservation/collection.html?period=&material=&detail=795301f4-443e-44b7-94c9-c87f236acff8

[5] https://github.com/udk-ale/AlgorithmicMuse

[6] https://gymnasium.farama.org/api/env/

"draw" or to "move". If it is "draw" a line will be drawn from the current to the next point. In that same logic, strokes can be extended endlessly using bezier interpolation between all active points. A "move" action interrupts the active stroke and the brush moves without painting. This would give the environment a total of 48 actions (24 draw and 24 move). Tests in the learning process revealed reducing the move actions to only the smaller radius accelerated learning as it reduced the action space without loss in drawing abilities (see Figure 1). For the study images were generated with 50, 100, 150 and 200 total actions.

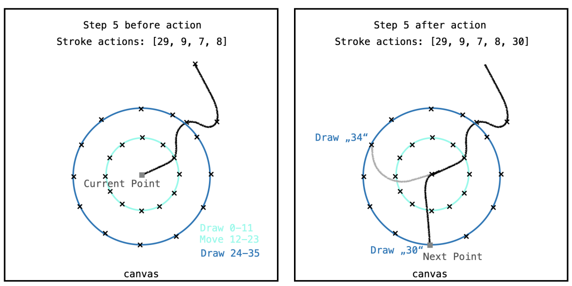


Figure 1: *Possible actions in the training environment, showing observation before and after action taken*

The observation space is a 640 x 640 pixel array in which the agent sees the current point as a rectangle of 9x9 grey pixels and the canvas with all previously drawn strokes. A frame-stacked observation of the last four steps was considered, but it did not show any improvements in the Painting Models' training.

III. Reward function

The reward is calculated in every timestep after the action is executed. The first part of the reward formula consists of the change δ in the probability value (0.0 to 1.0) the classifier is giving based on how much it looks like the trained "class 1" of artworks. The second part is a coverage map checking how many zones are painted and punishing if the canvas is not explored enough by the agent. For the classifier, we use the computer vision model yolov8n-cls.pt[7], a pre-trained image classifier trained on a custom dataset containing the following classes, Class 1: finished artwork; Class 2: snippets of artwork; Class 3: random model actions; and Class 4: bad performing painting-models (see Figure 2).

$$reward_{step} = \frac{(\delta_{class1} \times 20) + \delta_{class2}}{20} - coverage$$

$$\delta_{class1} = proba_{class1,step_i} - proba_{class1,step_{i-1}}$$

$$\delta_{class2} = proba_{class2,step_i} - proba_{class2,step_{i-1}}$$

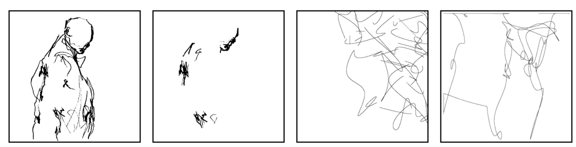


Figure 2: *Custom Dataset containing following classes: Class 1: finished artwork; Class 2: snippets of artwork; Class 3: random model actions; and Class 4: bad performing painting-models*

Figure 3 shows each action's reward over the 200 steps of one drawing, and Figure 4 displays the probability of Class 1: finished artwork and Class 2: snippets of artwork. The concept is similar to a GANs architecture where a discriminator evaluates the output of a generator [3], except the classifier here is used to calculate the change in class probability value from step(i-1) to step(i) (δ_{class1}). The idea is to give an appropriate reward value to a specific action and how much it influenced the appearance of a drawing tied to a step and observation for the Neural Network to learn.

[7] `https://docs.ultralytics.com/de/tasks/classify/`

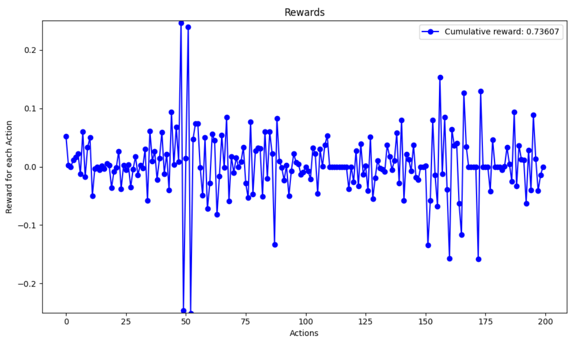


Figure 3: *Rewards of each action, the change of probability in Figure 4 prob of step(i) + prob of step(i+1) from -1.0 to 1.0*

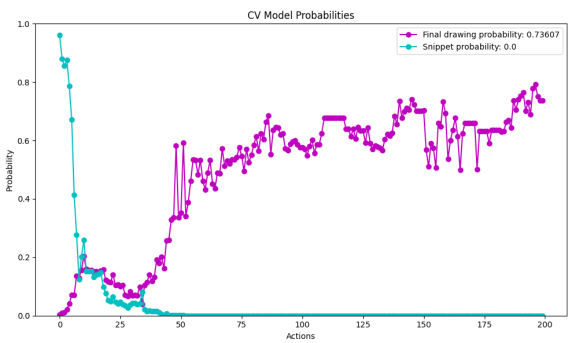


Figure 4: *Classifier calculating probability of Class 1: finished artwork (purple) and Class 2: snippets of artwork (blue) from 0.0 to 1.0*

The final score ranging from -1.0 to 1.0 is essential to evaluate the model's performance and primarily influences the learning process. To further emphasize the meanings of all possible scores, the following explanation helps to understand them. The final score is the cumulative reward from each step calculated by the previously explained reward function:

$$FinalScore = \sum_{i=0}^{total\ actions} reward_i$$

The concept of score normalization by Antonin Raffin [1] was utilized and helps to evaluate a model's performance compared to other RL problems. A score of 1.0 means the model achieved its goal; in this environment, the Agent draws good art like a human; no higher score is achievable. A final score around 0.0 would mean the Agent either does random actions because Class 3 random model actions are detected at a probability of 1.0, making Class 1 = 0.0 and Class 2 = 0.0. Alternatively, a bad policy, meaning Class 4, is detected at 1.0. If the final score, for example, is 0.74, it means the artwork is 74 Percent like the art from Class 1 (If we assume coverage punishment is 0.0).

Class 1 finished artwork has a factor of 20 over Class 2 snippet of artwork. This is useful to incentivize the model at the beginning of the drawing process to reward small steps and give a hint to the right direction because the gap between random actions and a finished artwork is just too big. For a good drawing, action rewards are likely to be from 0.01 to 0.1. It's the sum of multiple actions representing several strokes that creates a finished artwork with a cumulative reward/final score of 0.8 to 1.0.

The coverage punishment is a crucial component designed to intrinsically motivate [16] the Agent during training to explore more of the canvas than just a tiny part. In the training process, it is possible to observe that the Agent learned a policy where it hits the edge or corner of the canvas and rests on the already accumulated rewards. The model can not recover from this local minimum even with a low final score of 0.02. The coverage punishment is a countermeasure that for every ten steps, the Agent needs to have explored a certain amount of the 32 zones of the canvas. This amount is a parameter that can force a behaviour preferring the bigger picture rather than detailing at an early timestep. Failure to do so results in a proportional punishment to the number of steps taken. The worst case would be the model always taking a "move" action, meaning the canvas is not painted, leading to a total punishment and a final score of -1.0.

IV. Optimization method(s)

I. Proximal Policy Optimization

Among RL - algorithms, PPO is favored due to its actor-critic architecture [13], where the Agent is focused on finding a policy that predicts the best action rather than only optimizing the predicted reward values for all actions as a DQN would do. As the action space consisting of 36 discrete actions is large, after 50 actions, there are 10^{77} possible actions. Other research has shown PPO to perform better than other algorithms in large action spaces by updating the weights multiple times on the same batch of collected environment samples [12]. The used implementation is the Stable-Baselines3 PPO implementation [8], with standard settings, a learning rate of 0.002, n-steps: 1024 and batch-size: 64 for a total of 400.000 timesteps.

II. Simple tree search

A tree search is also suitable for finding solutions to generate artwork in the drawing environment. This tree search randomly looks at 12 of 24 „draw“ actions and selects the one with the highest positive reward. If there are no positive rewards, with a 0.61 probability, a random „move“ action is selected, and the reward is always 0.0 because there are no changes in the observation. To prevent the tree search from getting stuck in an endless loop of ”move” actions, as all ”draw” actions lead to a negative reward, there is a 0.39 probability of a random ”draw” action being selected. Thinking about this as ”the game of drawing” holds an interesting analogy to the game of chess: The possibilities of unique drawings and unique chess games are extremely high after just a few turns. Therefore, brute forcing the problem and looking at every possible outcome isn’t feasible. Alternatives like Monte Carlo Tree Search [17], and RL [15] in chess are standard practices for solving it.

[8] https://stablebaselines3.readthedocs.io/en/master/modules/ppo.html

V. Data

The dataset used for training the classifier (Yolo v8) Class 1, ”finished artwork,” consists of 600 sketches of human figures. Initially, the huggan/wikiart dataset[9] was considered. However, a custom dataset was created due to significant disparities between huggan/wikiart dataset and the custom environments style. These custom drawings, executed by human artists using pens or small brushes, emphasize a specific style of lines that portray figurative human drawings. The raw input for this dataset was modified to ensure its adaptability to the drawing environment. It was transformed to include only black-and-white pixels, as the environment produces images exclusively with black strokes. The dataset was further augmented by adding images with four different thresholds, simulating a gradual process of strokes in the picture. This practical approach yielded 2400 training images for the Yolo v8 model, which were subsequently divided into 75% training data and 25% validation data.

Further preprocessing involved removing noise, such as points and minimal lines, simplifying the input drawings. Additionally, for Class 2, which comprises snippets of artwork, new images were generated by randomly creating boxes containing some lines. These images contained between one and five boxes (see Figure 2).

These two classes represent the criteria the Painting Model is rewarded to resemble. Lastly, a class representing random noise was created, encompassing a range of 10 to 200 actions. Another class, ”bad models,” contains the initial training iterations of the Proximal Policy Optimization (PPO) algorithm. Iteratively updating the ”bad models” class with the best version facilitates Algorithmic Muse’s convergence towards the input data as it continually strives to surpass its previous iterations.

[9] https://huggingface.co/datasets/huggan/wikiart

VI. Results

I. Algorithms

After training the model for 400.000 timesteps, making 80.000 drawings per run, the model stagnates at around a mean total reward of 0.18. Further testing with slight hyperparameter changes validates a stagnation around 0.1-0.2 if any learning occurs. Since RL algorithms are susceptible to hyperparameters [2] and environment settings [12] and only start to converge after a very high sample size in the millions, this research's 10 training runs of 300k timesteps is not enough data to determine whether maximum results are possible [9]. Hyperparameter tuning and more extended training could solve the issue of early stagnation.

In contrast, the results of the tree search show that good results are achievable with the Agent's actions and current limitations in drawing complexity. With a breadth of 12 random actions and a depth of 1 action for drawings with 50 total actions, the search gets average results of 0.32. The distribution of results varies a lot. Being able to get results from 0.8 to 1.0, there are also many results of around 0.0. The cause may be the search taking a route at the beginning of the drawing process, from which it cannot recover as it is limited to only looking at a depth of 1 branch. Subjectively good results were obtained from this approach to compare them further in the study against human artists.

II. Study

The study involving 203 subjects compared 80 images generated by the tree search of Algorithmic Muse. 20 images by human artists are included that mimic the stroke-based line style of Algorithmic Muse's drawing environment and 20 images generated with random actions by the Agent. The categories of 80 generated images are further classified into A1, A2, N1, and N2 (A = AI and N = Random), each with distinct characteristics. For instance, the first category represents model-making drawings continuously drawn on canvas, known as "one-line drawings." The second category allows the Agent to "move" without the pen touching the canvas, resulting in multiple visible strokes. These categories were numbered, and the ending number represents how many actions they consist of: A1-XY-50, A1-XY-100, A1-XY-150, and A1-XY-200. This logic applies to A2, N1, and N2 (See appendix Figure 13).

In Part 1 of the study, the subjects had to choose two images out of 12 they liked the most. These 12 images consist of 4 A1, 4 A2, 1 N1, 1 N2, and 2 human artist drawings. All 120 competed against each other in 10 groups, which did not change during the study.

Part 2 consisted of subjects evaluating drawings on a scale of 1 to 7. The lowest value was assigned to the word "ugly," 4 to "mid," and 7 to "beautiful." The exact half of the 120 images were being evaluated.

Part 3 was optional. People could choose the one they liked the most from the 80 Algorithmic Muse images, and the robotic arm would paint it for them as a reward for completing the study.

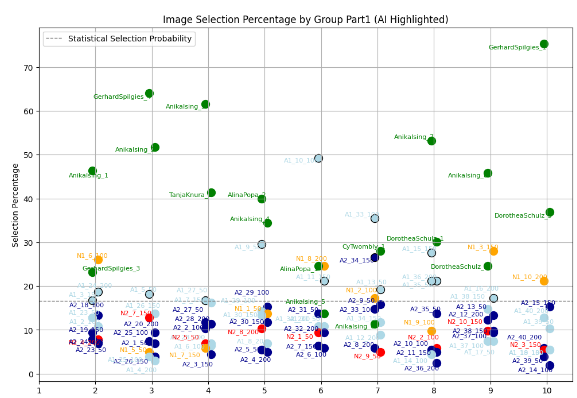


Figure 5: *Image Selection Percentage by Group Part 1 (AI Highlighted): A1 (light blue), A2 (dark blue), N1 (orange), N2 (red), and Human Artists (green). Human drawing has the highest selection rate for 8/10 groups, but 9 AI images have a higher selection rate than at least one human drawing in their group*

The results from Part 1 are shown in Figure 5 and demonstrate a general preference for the drawings by human artists. In 8 of 10 groups, one of the two human drawings had the highest selection percentage. In 5 of them, both human drawings were selected most often. Part 1 of the study leads to overall average selection percentages of 38.9% Human artists, N1 17.3%, A1 14.8%, A2 9.4%, and N2 8.4%. Human artists highly dominated groups 3 and 10. In contrast, Groups 6 and 7 had higher selection percentages of AI images. In total, 9 images by Algorithmic Muse had a higher selection percentage than a human drawing in its respective group.

Figure 7 can be used to interpret possible reasons why some human images were dominant and ranked higher than Images produced by Algorithmic Muse in Part 2. Human Artist's drawings in this study had a higher degree of detail and unique style and were made with pen on paper. It can be subjectively observed that better-performing images, in all categories, generally have more figurative and human articulation in their drawings. The more abstract the artist/Algorithmic Muse draws, the lower the selection percentage in Part 1, and a lower score was measured in Part 2 (comparing GerhardSpilgies-4 to AnikaIsing-4 and A1-10-100 to A1-12-200).

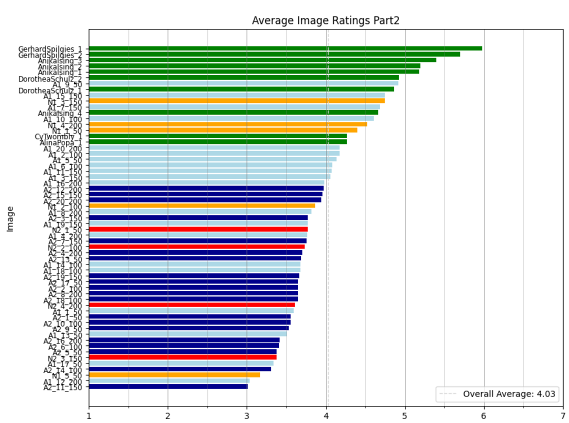


Figure 6: *Average Image Ratings of Part2. Human artists scoring the highest followed by A1 and N1 drawings*

Consistent preferences in drawing selection by participants could be proven throughout the study. Spearman's rank correlation revealed a robust and positive correlation (r = 0.79, p ¡ 0.01 see appendix Figure 9) between image selection frequencies in Part 1 and average ratings in Part 2. Ranking the images by their ratings shows that the Human Artist's drawing ranked highest, with some AI Images beating the Human's Score the same way as in Part 1 (see Figure 6).

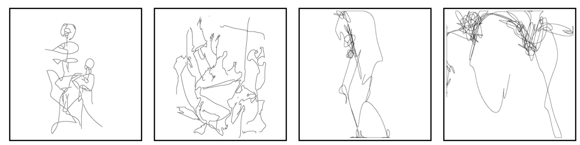


Figure 7: *Images from the study (left to right): GerhardSpilgies-4, AnikaIsing-4, A1-10-100 and A1-12-200.*

In conclusion, this study shows that drawings by Algorithmic Muse can perform comparably to Human artists and, in some comparisons, even surpass them. Even though a higher preference towards human images in this study was measured, images with the A1 and N1 "one line drawings" style achieved notable success. Converting the findings and rankings of this study into a new style, "A3," could improve the perceived appearance of drawings by a human audience. Key aspects would be a degree of abstraction, human mannerisms, and drawing techniques. Consistency in scores through all Parts of questions further validates the reliability of the data.

VII. Where to go from here ?

Optimizing the two algorithmic approaches used in the image drawing process presents an intriguing area for further developments. The current simple tree search algorithm lacks depth, and it would be beneficial to explore how the image quality could improve by comparing a series of actions leading to a complete brushstroke with alternative brushstrokes. Drawing inspiration from chess, various solution-finding methods, such as a Monte Carlo Tree Search or a hybrid approach combining DeepQ-Networks with tree search, could offer novel ways to generate brush strokes.

While Proximal Policy Optimization did not yield outstanding results, initial learning progress shows promise, suggesting a potential for fast, intuitive decision-making when tree search proves too time-consuming. However, hyperparameters and the reward function require further tuning.

Translating this digital algorithmic approach to art creation into the physical world is what inspired this research. This system can be seen as an independent artist on its own by connecting Algorithmic Muse to a robotic arm that paints. The resulting artworks, born from algorithms calculating and creating, are similar to the human artistic processes. The next stage involves collaboration: As Algorithmic Muse observes the canvas and makes decisions based on it, there is a unique opportunity for human artists to interact with this system, alternately influencing each other's creative perceptions in a shared artistic endeavor.

VIII. Acknowledgments

I extend my sincere gratitude to Prof. Dr. Daniel D. Hromada and Dipl. Ing. Christian Schmidts for their invaluable guidance and expertise throughout this project. Their support has been pivotal in its realization. I also wish to thank my peers in the Design and Computation program for their collaborative spirit and mutual support, which have greatly enriched this experience. Their collective insight has been a cornerstone of my project's success. Additionally, I extend special thanks to Prof. Oliver Fritz for his knowledge, inspiration, and assis- tance during my bachelor's degree at HTWG Konstanz.

References

[1] Antonin$_{R}affin$. *Rliable: Better Evaluation for Reinforcement Learning - A Visual Explanation*. Apr. 2, 2024. URL: https://araffin.github.io/post/rliable/ (visited on 04/02/2024).

[2] Theresa Eimer, Marius Lindauer, and Roberta Raileanu. *Hyperparameters in Reinforcement Learning and How To Tune Them*. 2023. arXiv: 2306.01324 [cs.LG].

[3] Ian J. Goodfellow et al. *Generative Adversarial Networks*. 2014. arXiv: 1406.2661 [stat.ML].

[4] Zhewei Huang, Wen Heng, and Shuchang Zhou. *Learning to Paint With Model-based Deep Reinforcement Learning*. 2019. arXiv: 1903.04411 [cs.CV].

[5] Biao Jia et al. *PaintBot: A Reinforcement Learning Approach for Natural Media Painting*. 2019. arXiv: 1904.02201 [cs.CV].

[6] Timothy P. Lillicrap et al. *Continuous control with deep reinforcement learning*. 2019. arXiv: 1509.02971 [cs.LG].

[7] Bingchen Liu, Kunpeng Song, and Ahmed Elgammal. *Sketch-to-Art: Synthesizing Stylized Art Images From Sketches*. 2020. arXiv: 2002.12888 [cs.CV].

[8] Jonas Oppenlaender. "A taxonomy of prompt modifiers for text-to-image generation". In: *Behaviour amp; Information Technology* (Nov. 2023), pp. 1–14. ISSN: 1362-3001. DOI: 10.1080/0144929x.2023.2286532. URL: http://dx.doi.org/10.1080/0144929X.2023.2286532.

[9] Sohum Padhye. *Building a Reinforcement Learning Agent that can Play Rocket League*. Apr. 2, 2024. URL: https://sohum-padhye.medium.com/building-a-reinforcement-learning-agent-that-can-play-rocket-league-5df59c69b1f5 (visited on 04/02/2024).

[10] Aditya Ramesh et al. *Hierarchical Text-Conditional Image Generation with CLIP Latents*. 2022. arXiv: 2204.06125 [cs.CV].

[11] Ricardo Andres Diaz Rincon. *Generating Music and Generative Art from Brain activity*. 2021. arXiv: `2108.04316 [cs.HC]`.

[12] John Schulman. *Deep RL Bootcamp Lecture 5: Natural Policy Gradients, TRPO, PPO - lecture at Berkley*. Apr. 2, 2024. URL: `https://www.youtube.com/watch?v=xvRrgxcpaHY&ab_channel=AIPrism` (visited on 04/02/2024).

[13] John Schulman et al. *Proximal Policy Optimization Algorithms*. 2017. arXiv: `1707.06347 [cs.LG]`.

[14] Sakib Shahriar. *GAN Computers Generate Arts? A Survey on Visual Arts, Music, and Literary Text Generation using Generative Adversarial Network*. 2021. arXiv: `2108.03857 [cs.AI]`.

[15] David Silver et al. *Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm*. 2017. arXiv: `1712.01815 [cs.AI]`.

[16] Satinder Singh et al. *Intrinsically Motivated Reinforcement Learning: An Evolutionary Perspective*. Apr. 2, 2024. URL: `https://web.eecs.umich.edu/~baveja/Papers/IMRLIEEETAMDFinal.pdf` (visited on 04/02/2024).

[17] Maciej Świechowski et al. "Monte Carlo Tree Search: a review of recent modifications and applications". In: *Artificial Intelligence Review* 56.3 (July 2022), pp. 2497–2562. ISSN: 1573-7462. DOI: `10.1007/s10462-022-10228-y`. URL: `http://dx.doi.org/10.1007/s10462-022-10228-y`.

IX. Appendix

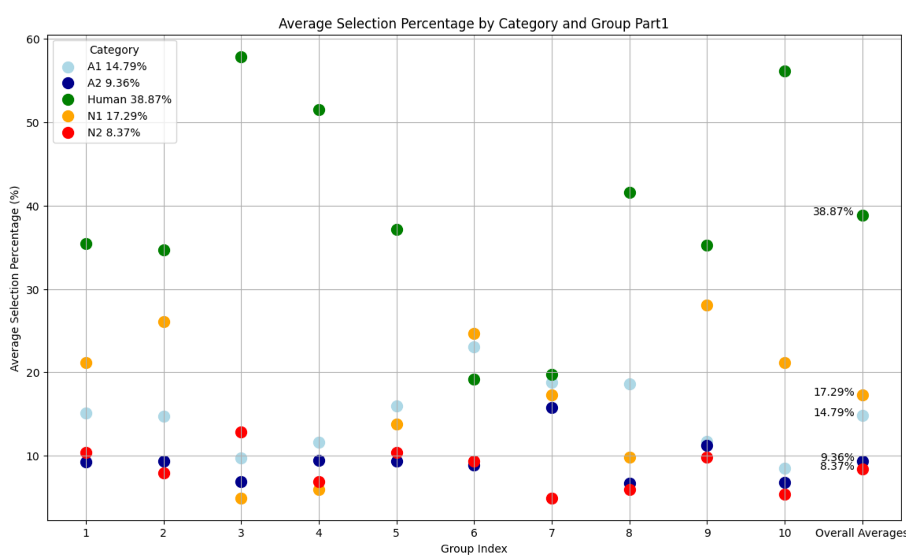


Figure 8: *Average Selection Percentage by Category and Group in Part1*

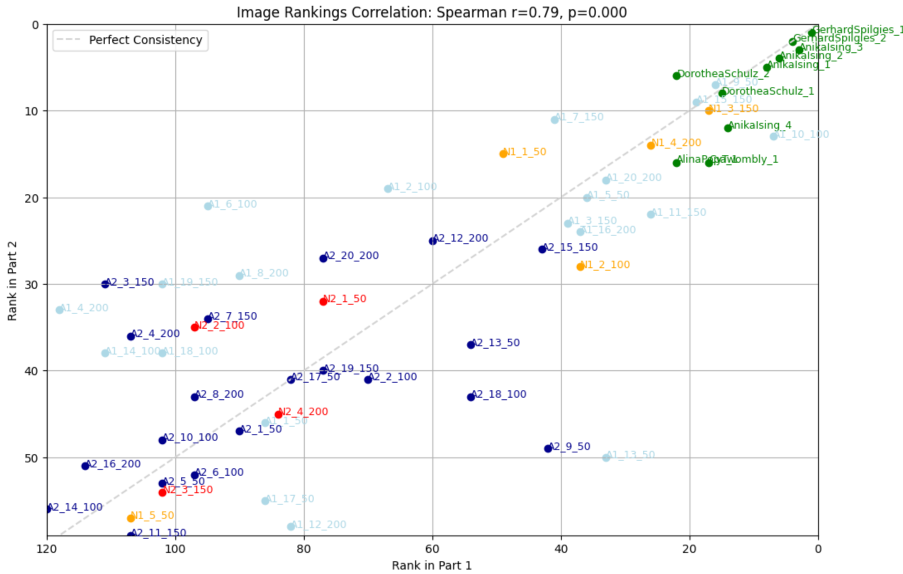


Figure 9: *Image Rankings Correlation: Spearman r=0.79 and p=0.000 revealed a robust and positive correlation between Part 1 and Part 2*

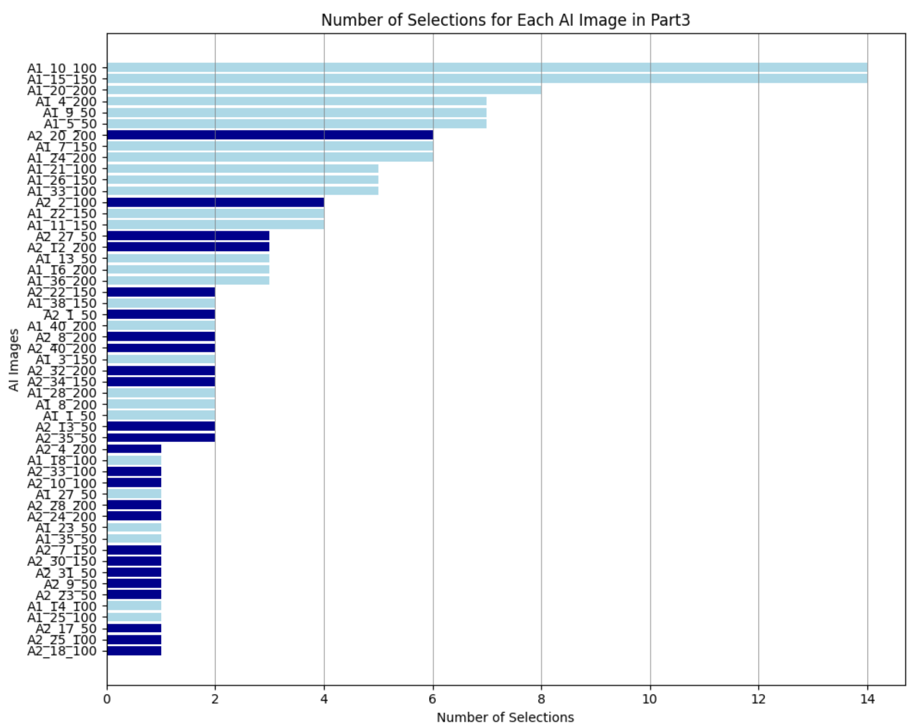


Figure 10: *Number of Selections for each AI Image in Part 3 showing the preference of the style A1 over A2. 169 participants will receive the selected painting drawn by the robotic arm*

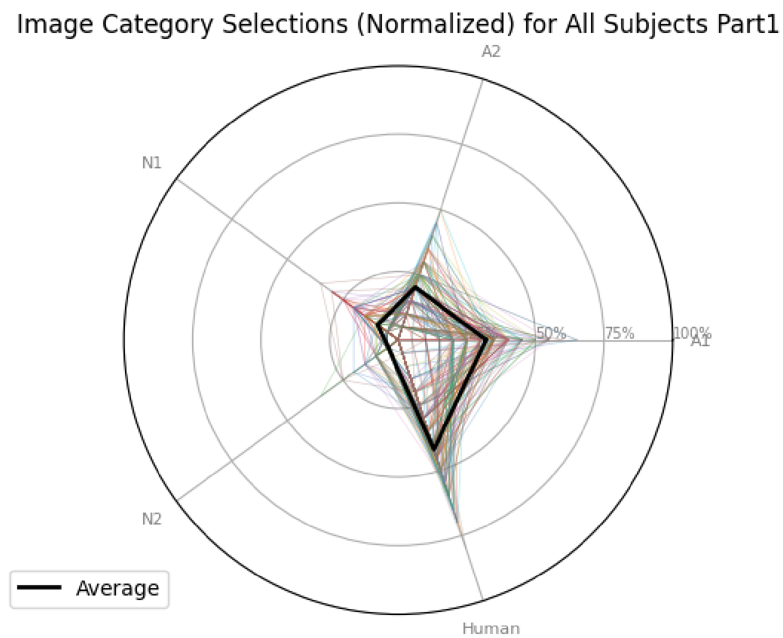


Figure 11: *Image Category Selections (Normalized) for All Subjects Part1 demnostrates selections made by each participant. Average scores consist of participants' individual taste showing a strong preference for types of categories*

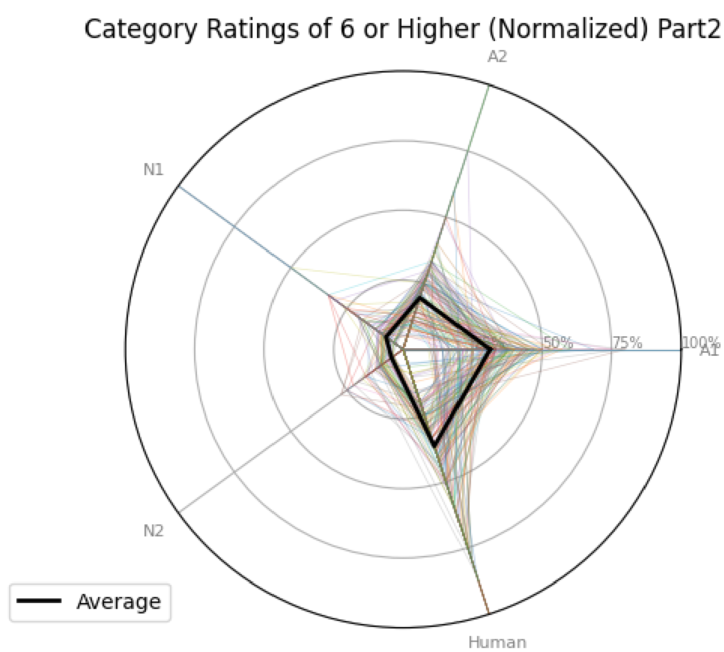


Figure 12: *Category Ratings of 6 or Higher (Normalized) Part2 showing individual participants preferences. Note similarities to Part 1 displayed in Fig11)*

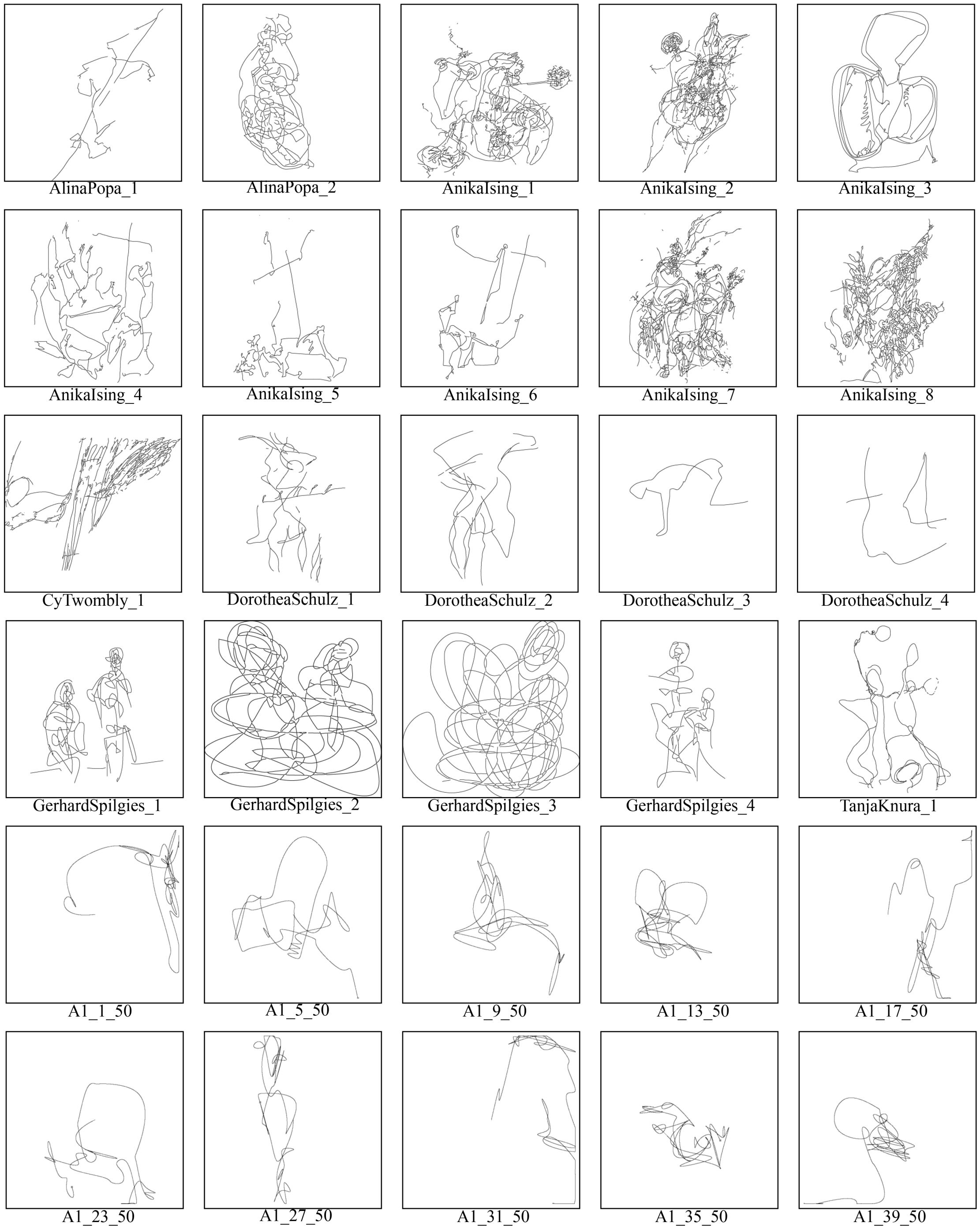


Figure 13: *All drawings in the study (1/4): Human and A1*

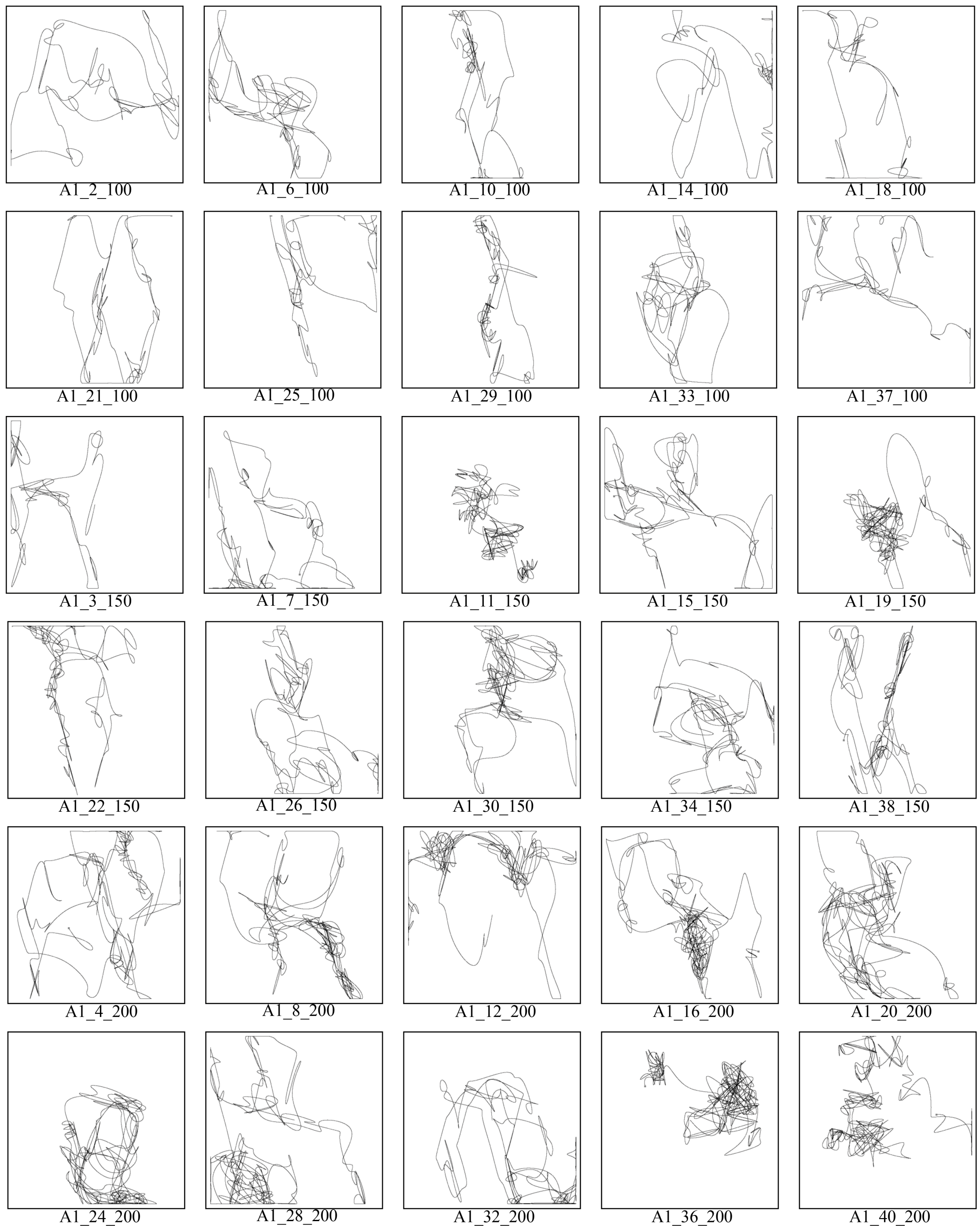


Figure 14: *All drawings in the study (1/4): A1*

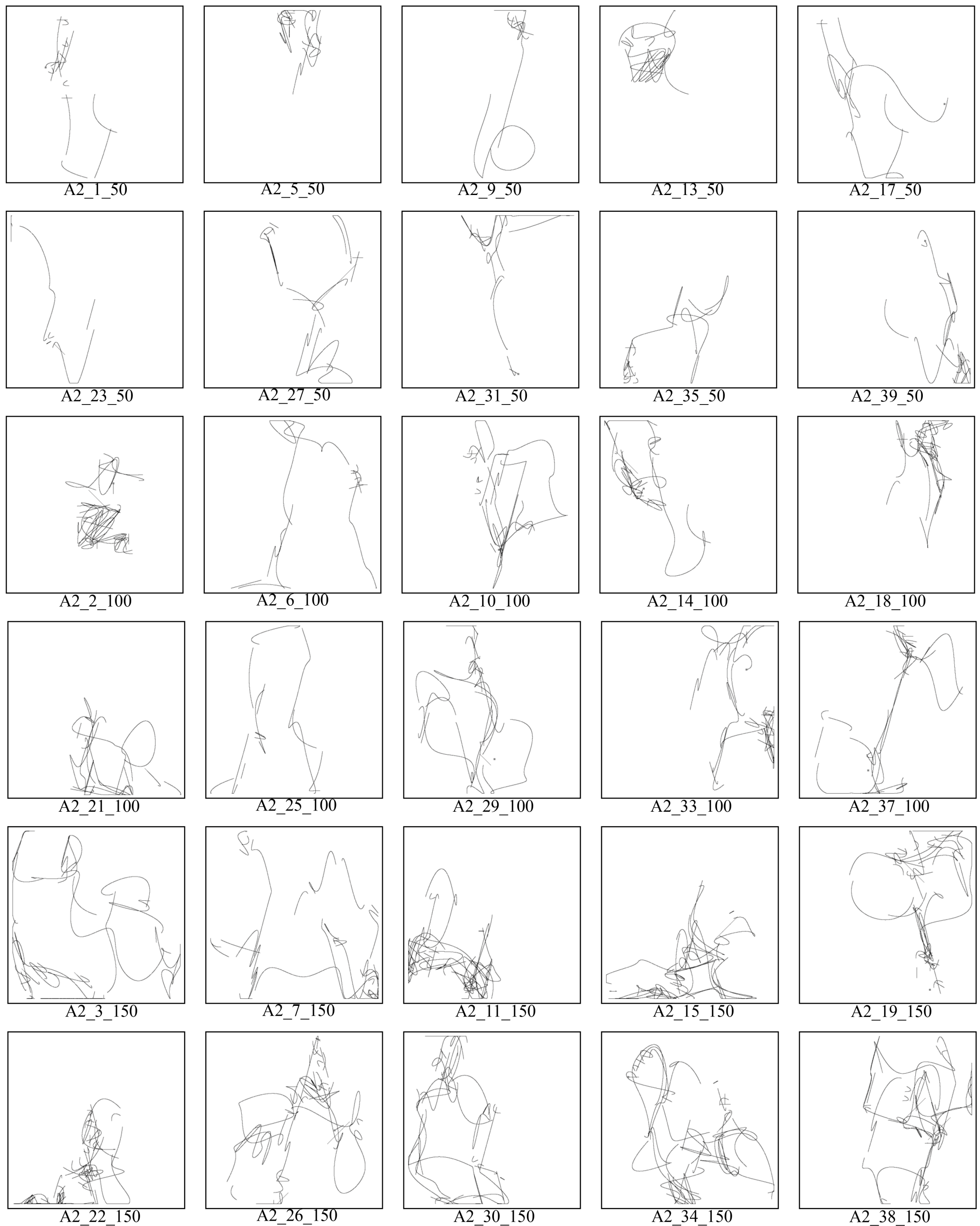


Figure 15: *All drawings in the study (1/4): A2*

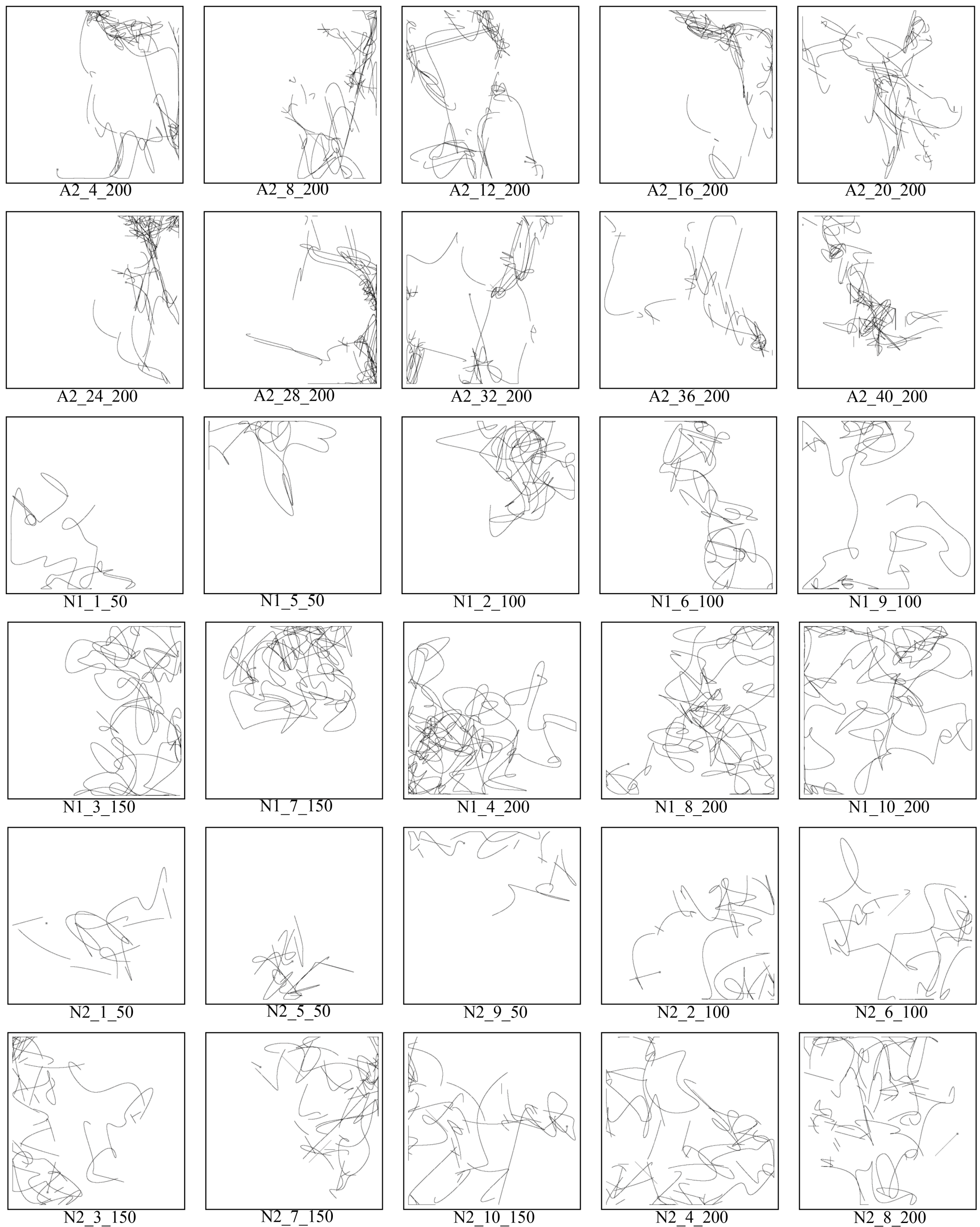


Figure 16: *All drawings in the study (4/4): A2, N1 and N2*