COMP3004 Designing Intelligent Agents

Name	Student ID
Khor Yong Teng	20089417

Link to code:

https://uniofnottm-my.sharepoint.com/personal/hfyyk4_nottingham_ac_uk/_layouts/15/onedrive.aspx?id=%2Fpersonal%2Fhfyyk4%5Fnottingham%5Fac%5Fuk%2FDocuments%2FCOMP3004%20Coursework%2FRL%5FDriving&ga=1

Reinforcement Learning on CarRacing-V0 with Different Image-Based Observation Spaces

Introduction

Reinforcement Learning is an area of machine learning that deals with enabling intelligent agents to learn behaviour through trial-and-error interactions within a dynamic environment [1]. Unlike the more commonly explored supervised machine learning that learns from a training set of labelled examples, reinforcement learning agents learn from their own experience, being provided a reward for each action determined by the defined goals within an environment. Agents are not told which actions to take, but instead must discover which actions yield the most reward by trying them [2], in an attempt to maximise the cumulative reward obtainable within the environment.

However, an issue that arises for reinforcement learning is its sensitivity to problem formulation in aspects including observation spaces, action spaces and the reward function, especially in cases of deep reinforcement learning. [3] Observation spaces may be designed based solely on prior knowledge and without established guidelines and principles. Hence, finding the optimal observation space is a significant design challenge that may heavily impact the training of reinforcement learning algorithms. This is increasingly so when it comes to image-based reinforcement learning, where the use of which oftentimes leads to the policy failing to converge [4].

Research Question

How does different image-based observation spaces affect the performance and behaviour of reinforcement learning agents trained in the same environment?

Aims and Objectives

The aim of this project is to investigate the effects of using different types of image-based observation spaces as input when training reinforcement learning agents within the same environment. Image observations involved will include different simplifications of the original RGB image, as well as other observation space augmentation methods that may improve the performance of the agent, such as framestacking and resizing.

To achieve this aim, the following key objectives will need to be met:

1. A reinforcement learning algorithm is trained on different observation spaces, producing agents able to be used for evaluation.
2. Agents evaluated to retrieve performance obtained at different timesteps of training for comparison.
3. Learned behaviour of agents within the environment are inspected and recorded.

Related Works

Image-based reinforcement learning is not new and has been explored in [4] in an attempt to solve OpenAI Fetch Robotic environments. Within the work, it was found that image-based reinforcement learning policies could perform almost as well as state based reinforcement learning policies, with only a minute difference in mean reward achieved, hypothesised to be due to blurry images thus decreasing the accuracy of the robot.

Investigation into observation space representation has been investigated as well. In [3], an analysis of different observation spaces was done where it was concluded that proper design of observation space matters. Observation space design can greatly improve or impair learning speeds within an environment, although the design of which may not generalise across all settings.

Within [5], research into image augmentation for reinforcement learning was done. A simple regularisation was implemented on the image observations, resulting in a significant improvement in SAC algorithm performance on continuous control tasks. It is also demonstrated that the method implemented is robust to the choice of hyper-parameters. This further shows the importance of observation space design on reinforcement learning performance, especially when images are involved. The importance of observation space design is also reflected in [6], where grayscaling, downsampling and framestacking of the observations were done during preprocessing, methods that are further investigated in this project.

Methodology

The approach towards the investigation can be mainly split into 4 parts, namely the environment setup, environment modifications, agent implementation and experimental setup, each with its own set of challenges and solutions. Code used for this investigation was implemented in python, being the one of the most versatile languages for machine learning tasks. For the ease of implementation for the various components, multiple libraries were used for designing the approach.

Environment Setup

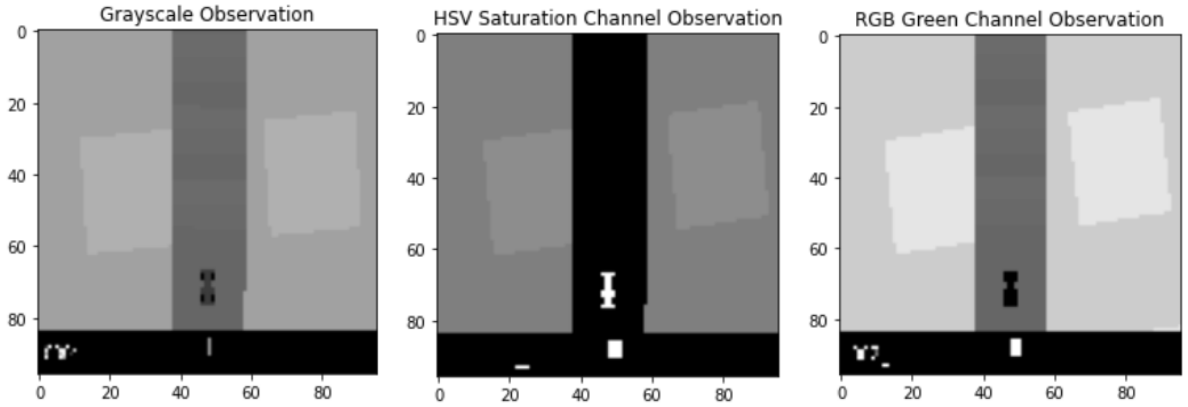
The CarRacing-V0 environment from OpenAI Gym [7] was decided to be used as the base and for this task due to its suitability being an image-based environment. The environment is a racing environment with a top-down view, where the track, consisting of tiles, are generated at random every episode, generalising the learning process and preventing learning through memorization by the agent. The original observation space is an RGB image, consisting of 96 x 96 pixels with 3 channels representing the RGB channels, resulting in a 96 x 96 x 3 input for the agent. The environment features a continuous action space for controlling the race car with 3 actions that can be set separately which are steering, gas and braking. The actions can have values ranging from -1 representing a full left to 1 representing a full right for steering, and 0 to 1 for gas and braking representing the strength of the action. The reward of the environment is calculated by -0.1 every frame and +1000/N for every tile visited in the track. The environment can be considered solved if a reward of 900 and above is achieved consistently.

Initial experimentation has shown that the environment would have issues with extremely long episode lengths, due to not hitting the episode end requirements of being out of bounds or finishing all tiles. This was fixed by modifying the environment and imposing a time limit of 2000 timesteps per episode. The shortened episode length improved the training process due to allowing the environment to reset much faster and not spend too much time on unsuccessful attempts.

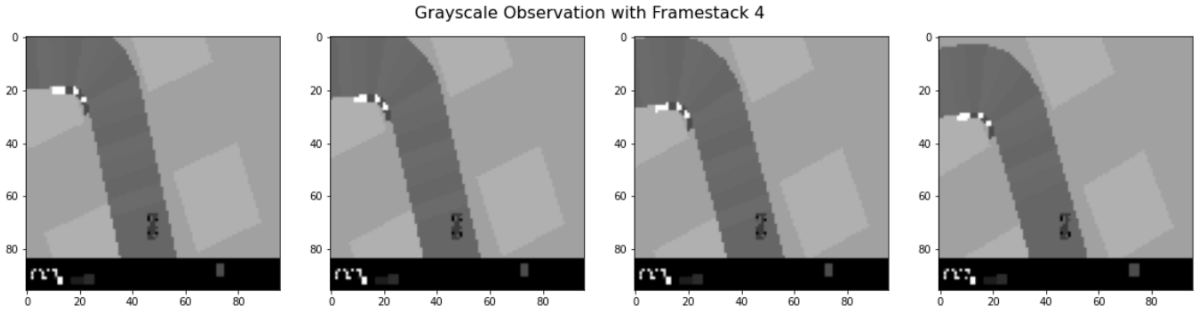
Environment Modifications

As the experiments require the use of different variations of the observation space, the SuperSuit [8] library was used for the modification of the CarRacing-V0 environment, acting as a wrapper to create environments with new image-based observation spaces.

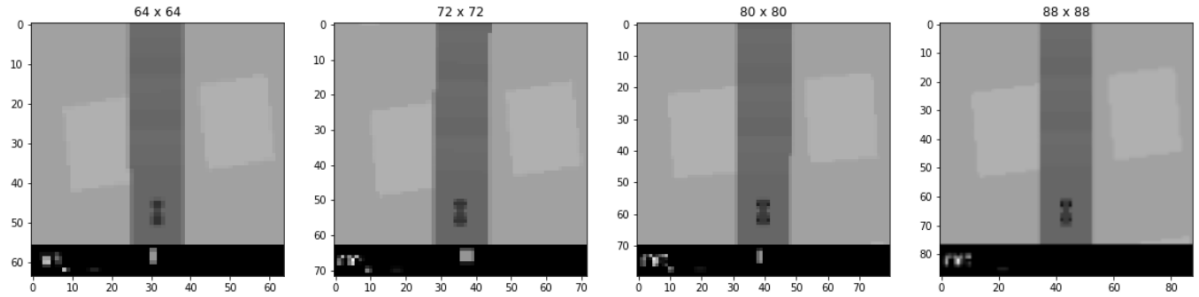
Firstly, a type of modification made was colour space conversion. The original RGB observations were converted into grayscale, RGB green channel and HSV saturation channel observation spaces shown below. The conversion also acts as a method for simplifying the observation space from 3 channels into 1, reducing the dimension to 96 x 96, in hopes of making the environment easier to learn, resulting in a higher reward. Aside from grayscaling which was typically implemented in other projects, the other conversion methods are tested as well to conclude their feasibility.



Secondly, frame stacking of the best performing observation from colour space conversion was implemented. Frame stacking is the act of stacking the previous frames of the observation space and returning it as a new observation space with an extra dimension with the size of the number of frames stacked. Frame stacking is said to have the capability of adding significant amounts of critical information regarding movement and velocity, and will be validated through the experimentation process. A modified environment was created to experiment with 2, 3, 4, and 5 framestacks. Below is an example of frame stacking grayscale observation by 4 frames.



Lastly, the final modification made will be downsampling the observation space. The resulting observation spaces are resized versions of the best performing framestack environment, with decreased height and width in terms of pixels. The size reduction is done through downsampling using linear interpolation where the new pixel values are derived. Observation spaces with image size of 64 x 64, 72 x 72, 80 x 80, 88 x 88 are used within the experiment to determine learning will be affected. Examples of the resizing are shown below.



Agent Implementation

According to the environment, the acting agent will be a race car that has a continuous action space with 3 actions that can be set separately which are steering, gas and braking. The goal of the agent within this environment is to complete the track as fast as possible to obtain the maximum possible reward, as defined by the reward calculation specified. The agent is rewarded for surpassing a tile and penalised for each timestep spent having not completed the track.

To allow the agent to learn through reinforcement learning, the stable-baselines3 library [9] was used to implement the reinforcement learning algorithm acting as the agent. Stable-baselines3 has a plethora of implemented algorithms made available, and the A2C algorithm, DDPG and PPO algorithms were experimented with prior to choosing an algorithm for the project. The PPO algorithm was chosen to be used in the end as it is very robust and showed decent performance straight out of the box. A2C and DDPG failed to converge and were not able to make any significant improvements within the environment.

Minimal configurations were done to the PPO algorithm prior to training. The policy network for the algorithm was set to “CnnPolicy” due to the observation space being image-based and the learning rate was set to 0.00003 to allow for more stable learning.

Experimental Setup

The experiment was done through iterative stages of training the PPO algorithm on modifications of the CarRacing-V0 environment relevant to a certain aspect of observation space design as stated in the “Environment Modification” section, followed by thorough evaluation, then choosing the best performing environment to act as the base for the next set of modifications.

As it was observed that each run of training might produce different results, even using the same observation space. 3 separate runs of training is done for each observation space such that the consistency of training can be observed. During the training phase, a checkpoint callback was implemented to save models at incrementations of 20000 time steps of training, with the maximum time step of each run being 1000000.

The evaluation of the models produced was done using the “evaluate_policy” function implemented in the stable_baselines3 library. Each model produced during a run of training with incrementing time steps trained, is evaluated using 10 episodes of the environment that is randomly generated, producing a mean reward and standard deviation representing the performance achieved. These statistics obtained are plotted using the matplotlib library into error band plots for visualisation. A plot is produced for each observation space, where the result of 3 runs are compiled together.

Lastly, the behaviour of the agent when using the different observation spaces was also observed and recorded. Behaviours of the best performing model using each of the observation spaces are placed the most focus. Videos of interesting behaviours are recorded as well as to present the findings.

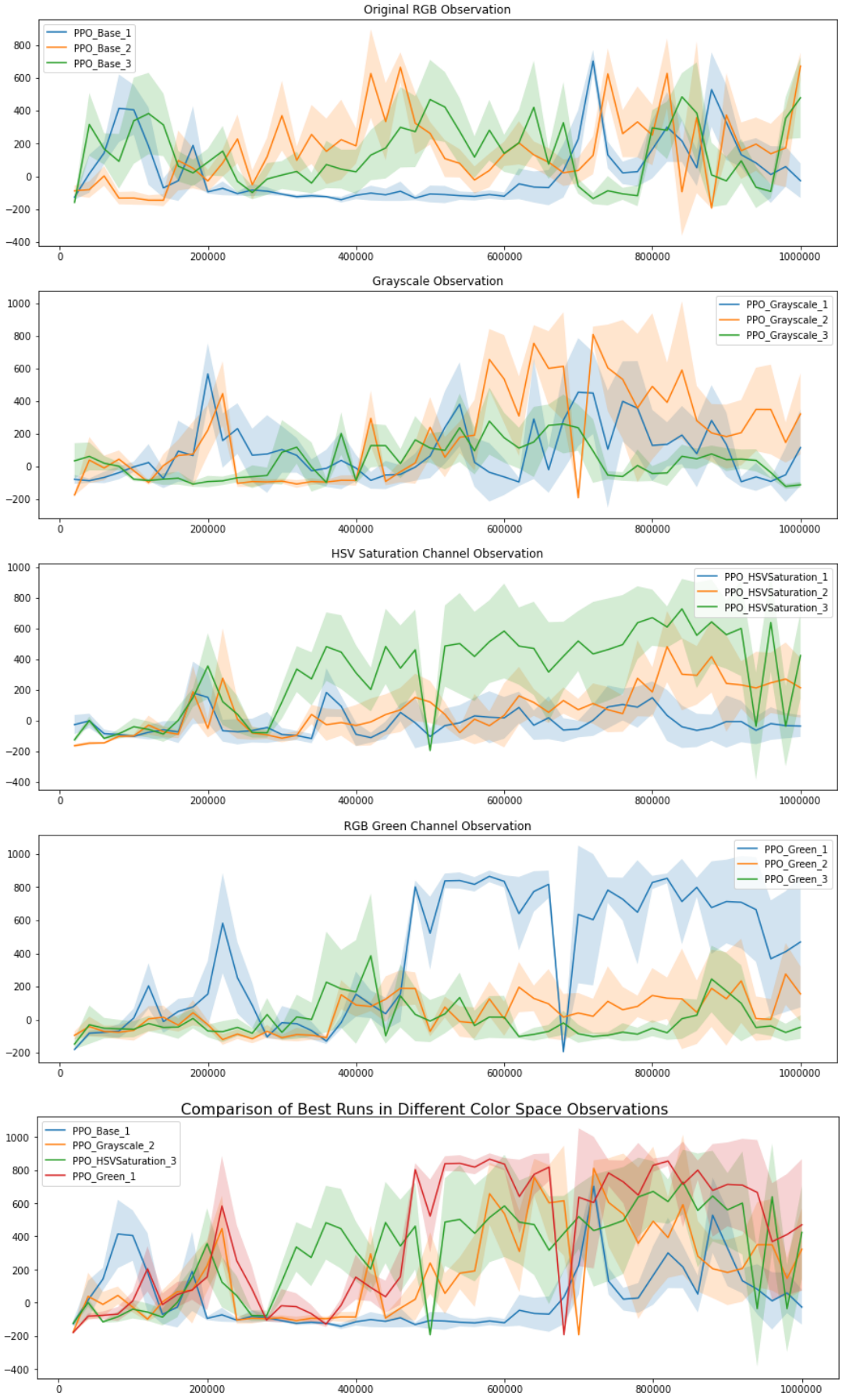
Results

Looking at all rounds of training done by the agent, the performance of the agents generally follows a trend of improvement in performance followed by deteriorating performance after the optimum is achieved for that run, which sometimes this trend is repeated, indicating agent relearning.

Colour Space Conversion

Reviewing colour space conversion as observation space simplification methods using RGB observations as a base, it can be observed that the best model all methods produced higher mean reward compared to RGB observations. The best mean reward was produced using the RGB green channel, followed by grayscale and then HSV saturation channel, with increasing standard deviations in reward in that order as well. Although RGB green channel produced the best model, its consistency in achieving good policies across different runs of training is poor as the mean reward of the other training runs never exceeded 400. Therefore, grayscale observation space can be seen as the overall best performing in terms of mean reward and consistency, and is used as the base of experimenting with frame stacking.

Mean Reward Achieved Against Timestep Trained with Different Color Space Observations



When using the original RGB observation space, the race car agent can be seen to have medium movement speed and make rapid left and right turns while moving normally. A number of issues can be observed as well such as a tendency to skip corners, go off track and be unable to recover, and occasionally get stuck and stop moving entirely until the episode ends. An interesting problem that appears as well is that the agent has trouble finding the right way at sharp corners, when 2 parts of the track are within view. The agent would move to the area between the 2 tracks and start spinning between them.

When observing the grayscale observation space, the agent moves more steadily and will still occasionally skip tracks but with lower frequency. It is also better at controlling when 2 tracks are within view, being able to keep control and continue the track after reorienting itself, leading to a higher mean reward overall.

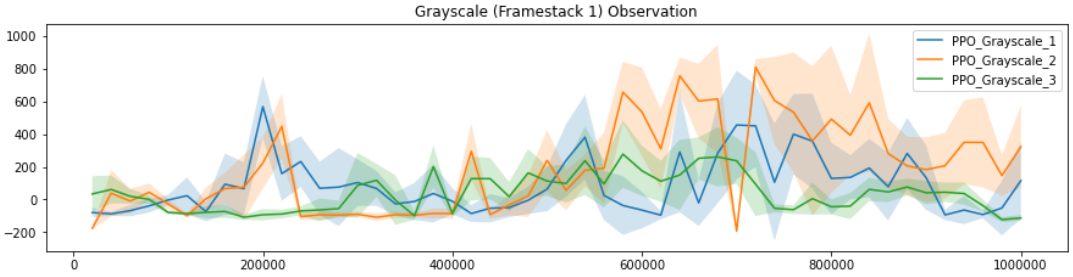
Using the HSV saturation channel, the agent no longer moves wobbly with rapid left and right turns while driving, however it does move at greater speeds, thus causing a higher chance to go off track. Going off track, the agent will try to reorientate itself but may start incorrectly moving in the opposite direction. When trying to join back onto the track, it might move very slowly at the edges when trying to centre itself. Getting stuck randomly is also an issue observed as with RGB observations. However, when using the HSV saturation channel, the agent did also develop the behaviour to break at certain corners, preventing it from going off track.

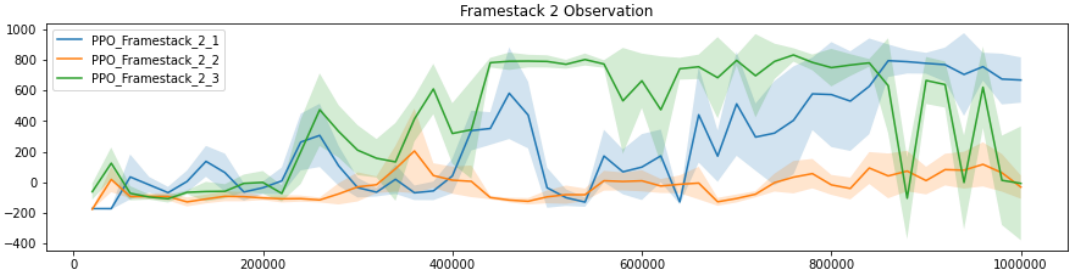
When the RGB green channel is used, the agent learns to move very wobbly, with larger rapid left and right turns when moving normally when compared to RGB observations, but although being wobbly, it is great at turning sharp corners while maintaining within the tracks. Compared to other observation spaces, it also skips very few tiles using this method. When presented with the same problem of having 2 tracks on screen, the agent will still go off track but is able to reorient itself and continue.

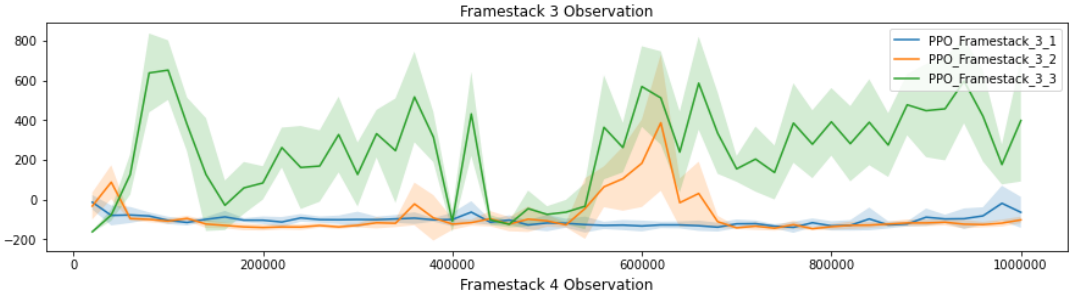
Frame Stacking

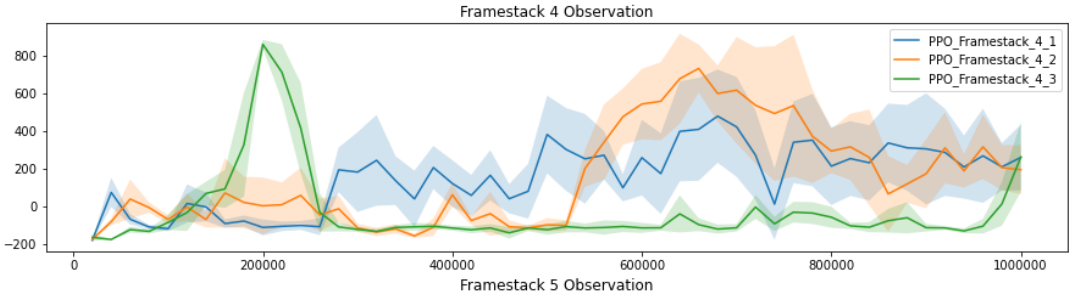
When the number of frames in the observation space are above 1, even numbers of framestacks showed significantly better mean reward achieved throughout all runs. Odd numbers of framestacks showed worse performance when compared to single frame grayscale observations. In the case of framestack 3, the policy is even unable to converge in one of the runs. Framestack 4 can be seen as the best observation space due to being able to achieve a high mean reward of 857.54 with a low standard deviation of 22.779649 in its best run and being able to do so in a small number of 200000 timesteps when compared to framestack 2. Although learning is quite stable for framestack 2, where a mean reward can be seen to be maintained at around 800 for a long time, the consistency of learning a decent policy in different training runs is lower than framestack 4.

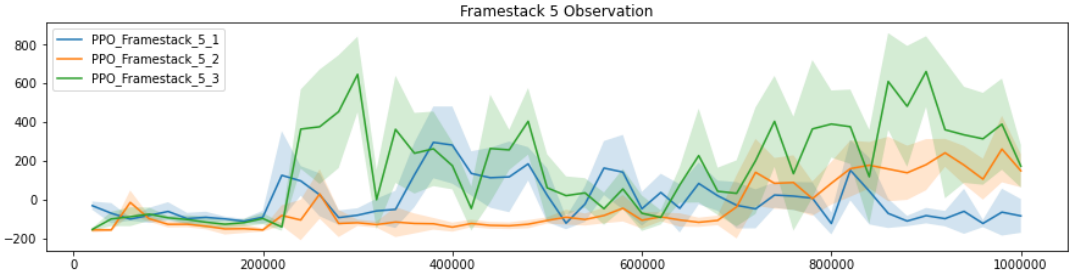
Mean Reward Achieved Against Timestep Trained with Different Framestacks

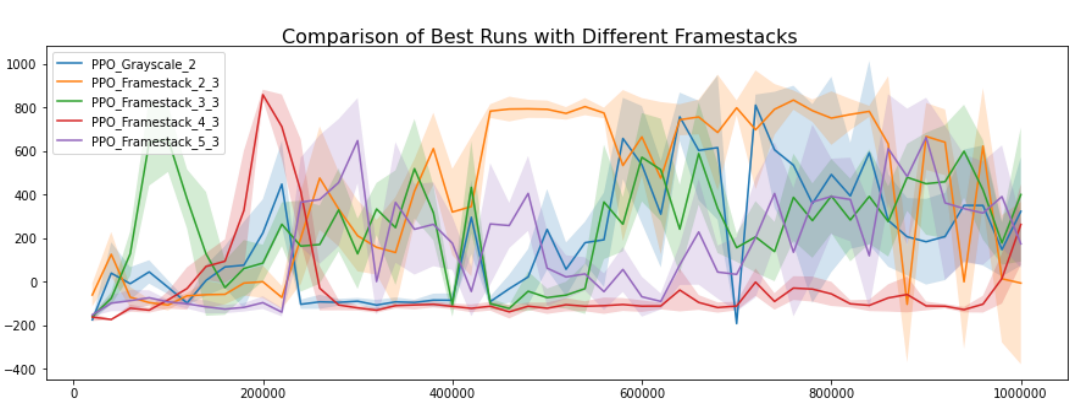












With all frame stacked observations, the agent moves in a wobbly fashion much like in RGB Green channel observation. Compared to framestack 3 and framestack 5, framestack 2 and 4 have much more beneficial behaviours that contribute to the mean reward.

Using framestack 2, the agent is able to complete the track occasionally. Good turns at corners can be seen most of the time, however the wobbly movement sometimes causes the agent to leave the track and re-enter it again, causing some tiles to be missed, especially at corners when it overcompensates the turn. The 2 track issue is still apparent using this observation space, causing the agent to move in the opposite direction after it reorients.

In framestack 3, the agent is even more wobbly compared to the other frame stacks, which causes the agent to leave the track even at straight sections. When leaving the track, the agent attempts to recover by rotating in a circle, back into the track, which sometimes works, but sometimes resulting in the agent moving in the opposite direction. The 2 track issue is present as well, causing it to skip corners.

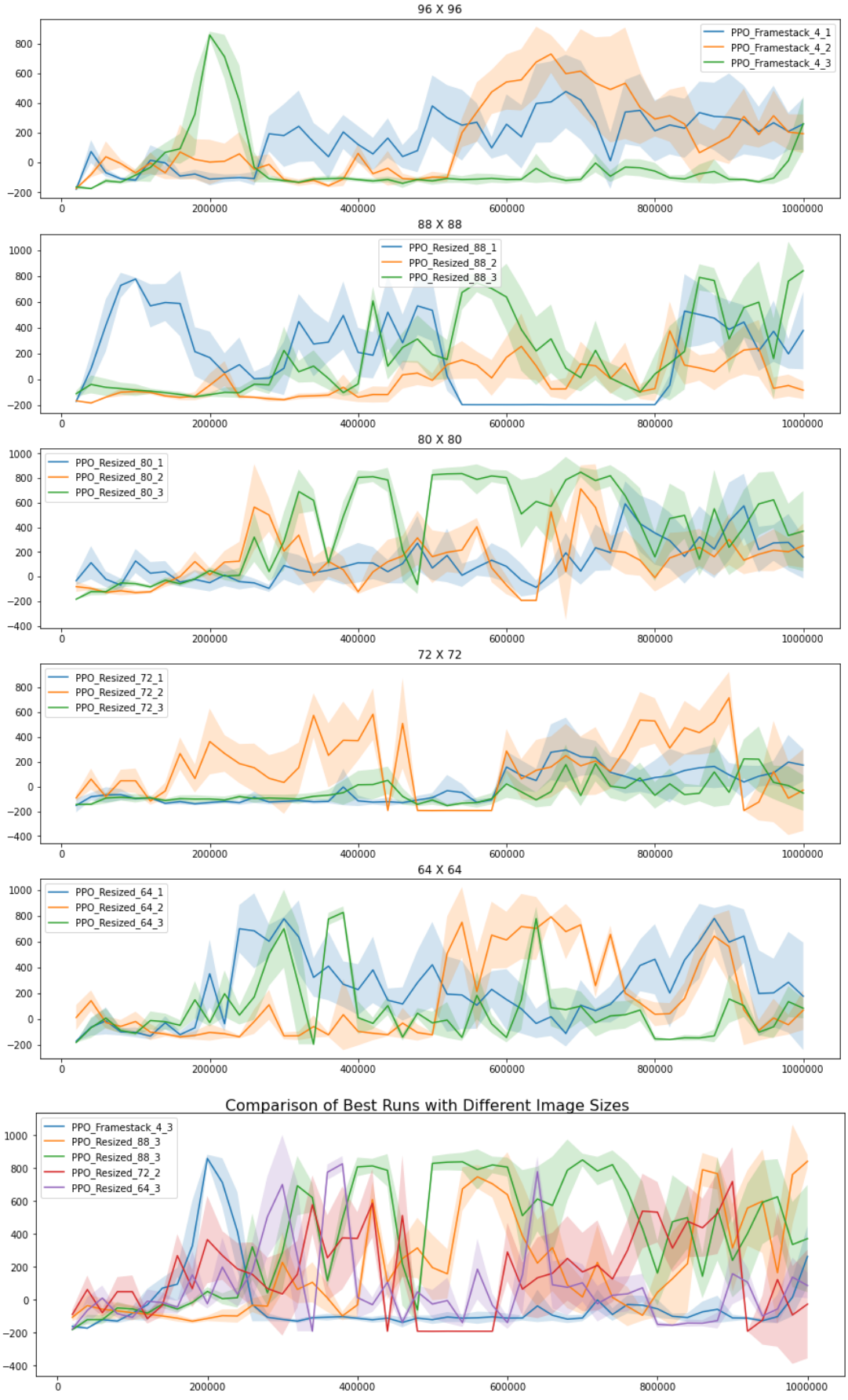
When observing the agent in framestack 4, the agent still wobbles but at a higher frequency with smaller amounts and moves at a slower speed, trading speed for consistency. It performs high quality turns but at especially sharp turns, it may leave the track. However, it is able to consistently recover in the right direction by rotating in a circle until it reaches the track again. Using framestack 4, the agent is able to complete the track very consistently, but due to lower speeds, it takes a longer time per episode, deducting more reward even though it can complete the track.

Framestack 5 appears to cause the agent to behave the worst. The agent using this observation space skips corners very frequently, and overcompensates during turns, causing it to leave the track. When leaving the track, it often gets stuck and stops moving completely. The 2 track problem also appears with framestack 5, where it gets stuck between the space between the 2 sections and spins uncontrollably.

Downsampling Results

Using the original 96 x 96 image size still produced the best performing agent within all downsampling results. However, 88 x 88, 80 x 80, and 64 x 64 image sizes still produced mean rewards similar to 96 x 96. 64 x 64 image size was able to produce better consistency in different training runs, producing a decent mean reward in all of its runs. 72 x 72 image size was only able to produce a maximum of 717.847107 mean reward within a run, where all other runs are unable to surpass a mean reward of 300.

Mean Reward Achieved Against Timestep Trained with Different Image Sizes



Comparison of Best Runs with Different Image Sizes

Agents in downsampled and resized observation spaces of 88 x 88, 80 x 80 and 64 x 64 showed very similar behaviour to the original framestack 4 (96 x 96) observation space acting as its base. However, some differences that can be observed is that corner skipping can be observed in 88 x 88, 80 x 80 and 64 x 64 observation spaces. In 64 x 64 as well, some skipping on straight tracks can be observed as well caused by wobbly movement near the edge of the track. 88 x 88 observation space can be seen as well to have lower amounts of wobble.

The agent 72 x 72 observation space however can be seen with a very wobbly movement, resulting in a lot of issues, including corner skipping. The overcompensation when turning causes corner skipping as well as the agent leaving the track, then continuing in the opposite direction. This turning into the opposite direction can also be seen to happen sequentially within a short duration of time.

Discussion

From the results, it can be observed that a grayscale observation with a framestack of 4 without downsampling performs the best for the CarRacing-V0 environment, which is in line with what was mentioned in [6], not including the downsampling used. Simplifying the colour space can be deemed as an effective method for improving the performance of reinforcement learning, however the type of colour space must be considered as different colour spaces do not lead to the same effectiveness. Different manipulations of the colour space as well leads to significant difference in behaviour in training the reinforcement learning agent even if the same environment is used.

Framestacking is also an important method to capture extra information required for better learning. From what was observed in the CarRacing-V0 environment, it leads to better ability for the agent to turn, possibly due to gaining the ability for capturing the turning speeds needed for adjustment. An interesting observation for frame stacking is that even numbers of frames seem to lead to a better performance. A possible hypothesis for this cause may be due to the network architecture employed by the PPO algorithm's "CNNPolicy", being much more effective for feature extraction when it comes to even numbers of frames.

Downsampling the observations might need to be done with care as some information might be lost with it. In this case, blurry borders might have led to the agent skipping tracks, a similar situation as mentioned in [4].

Conclusion

In this work, the effectiveness of certain image-based observation space manipulations was able to significantly improve performance of reinforcement learning within the CarRacing-V0, namely grayscaling the observations and frame stacking with 4 frames, aligned with approaches mentioned in previous works. Different observation space representations can greatly impact the learned behaviour of agents within the environment as well. Further work may be done to investigate if the results of this work may translate well to other environments, and if it may be representative of a general approach to image-based reinforcement learning tasks.

Link to Code

https://uniofnottm-my.sharepoint.com/personal/hfyyk4_nottingham_ac_uk/_layouts/15/onedrive.aspx?id=%2Fpersonal%2Fhfyyk4%5Fnottingham%5Fac%5Fuk%2FDocuments%2FCOMP3004%20Coursework%2FRL%5FDriving&ga=1

References

[1] Kaelbling, L. P., Littman, M. L., & Moore, A. W. (1996). Reinforcement Learning: A Survey. Journal of Artificial Intelligence Research, 4, 237–285. https://doi.org/10.1613/jair.301

[2] Sutton, R., & Barto, A. (2018). Reinforcement Learning An Introduction, Second Edition. The MIT Press, Cambridge, Massachusetts, London, England.

[3] J. T. Kim and S. Ha, "Observation Space Matters: Benchmark and Optimization Algorithm," 2021 IEEE International Conference on Robotics and Automation (ICRA), 2021, pp. 1527-1534, doi: 10.1109/ICRA48506.2021.9561019.

[4] Chahal, K. (2021, December 24). Image Based Reinforcement Learning - Karanbir Chahal. Medium. https://karanbirchahal.medium.com/image-based-reinforcement-learning-1a6da8eacd18

[5] Ilya, K., Dennis, Y., Rob, F. (2020). Image Augmentation Is All You Need: Regularizing Deep Reinforcement Learning from Pixels. arXiv:2004.13649 https://doi.org/10.48550/arXiv.2004.13649

[6] Volodymyr, M., Koray, K., David, S., Alex, G., Ioannis, A., Daan, W., Martin, R. (2013). Playing Atari with Deep Reinforcement Learning. DeepMind Technologies, London, England.

[7] Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., & Zaremba, W. (2016). Openai gym. ArXiv Preprint ArXiv:1606.01540.

[8] Terry, J. K., Black, B., Hari, A. (2020). SuperSuit: Simple Microwrappers for Reinforcement Learning Environments. ArXiv preprint arXiv:2008.08932

[9] Antonin R., Ashley H., Adam G., Anssi K., Maximilian E., Noah D. (2021). Stable-Baselines3: Reliable Reinforcement Learning Implementations. Journal of Machine Learning Research, 22(268), 1−8.