Enhancing Reinforcement Learning in Vision-Based Environments with Optical Flow

Amartya Mukherjee[1] **Jun Liu**[1]
[1] Department of Applied Mathematics, University of Waterloo
{a29mukhe,j.liu}@uwaterloo.ca

Abstract

Reinforcement learning (RL) has emerged as a powerful technique for training agents to excel in a wide range of sequential decision-making tasks, including playing video games in the Atari 2600 environment. While convolutional neural networks (CNNs) have been effective in extracting meaningful features from frames, the representation of motion remains a challenge. Optical flow (OF) gives information about the motion in sequential image data such as videos, which makes it useful in reinforcement learning. In this paper, we propose an approach to improve the performance of RL models in Atari environments by concatenating OF with raw image frames as input. Our experiments show that adding OF to an environment improves the training of the Deep Q Network (DQN) model and shows higher rewards compared to concatenating the present frame with its previous frame.

1 Introduction

In recent years, there has been a growing interest in reinforcement learning (RL) for a multitude of problems including optimal control and gaming. RL has shown promising capabilities in maximizing a notion of cumulative reward in environments with unknown dynamics through a balance of exploration in the environment and exploitation of the learned policies [1]. In RL tasks involving physical systems, understanding the dynamics of an environment is of paramount importance in finding an optimal policy.

The need for understanding the dynamics of an environment led to the advent of model-based RL (MBRL). One of the earliest works is the Dyna algorithm by [2], which uses a neural network to approximate transition and reward models. Another significant contribution to the field is by [3], which introduces an efficient algorithm to sample the value and promising actions from a search tree. This algorithm has been used by AlphaGo to train a Go player that outperforms humans [4].

In visual RL, agents must learn actions that maximize a notion of cumulative reward based on visual observations of the environment. The authors of [5] used RL to achieve optimal control tasks by observing the robot from a camera. They learn visual representations for goals, transition dynamics, and rewards to achieve this. The authors of [6] apply MBRL to Atari games. They use U-Net architecture to predict the next frame in a game based on the past four frames.

In the processing of sequential image data such as videos, motion vectors (MV) and optical flow (OF) have led to significant advances in video indexing and identification [7]. Due to taking significantly smaller values compared to images and being computationally efficient [8], MVs have aided in video compression [9]. Optical flow [10] is a variant of motion vectors under continuous position and time. It is obtained by solving a first-order partial differential equation (PDE)-contrained optimization problem. Both methods give estimates of the motion of videos, thus making them extremely useful in video understanding tasks.

In recent years, the application of motion vectors and optical flow in computer vision models has been of high interest to the ML community. The authors of [11] use optical flow as a visual representation to improve acoustic event detection models. The work of [12] trains a model to take a video and its motion vectors as input and outputs an optical flow. Furthermore, the work of [13] uses motion vector representations to assist in video super-resolution. Due to containing information on the velocity field of each frame, motion vectors were useful in increasing the frame rate in a video by assisting in inter-frame generation.

In this paper, we will focus on the application of OF to the training of vision-based RL models. For example, the environment could be an Atari game like Breakout or SpaceInvaders, where understanding the motion of the player and objects is crucial to scoring high points. We train a Deep Q Network (DQN) that takes the current frame of the environment with its OF. Since the OF is a visual representation that contains information about the motion in the environment, no such representation needs to be learned by the DQN. And OF is computed by solving a PDE-constrained optimization problem, which is significantly more computationally efficient compared to training a neural network and expecting it to learn these representations. We conduct a set of numerical experiments that show that integrating OF improves the training of the DQN and shows higher rewards compared to concatenating the present frame with its previous frame. We expect that incorporating optical flow will improve the performance of visual RL tasks in general.

2 Preliminaries

Throughout this work, we will refer to $I(x,y,t) : [0,I^{(x)}] \times [0,I^{(y)}] \times [0,\infty) \to \mathbb{R}$ as the (continuous time) image frame at time t at position (x,y). And we will refer to $\mathbf{I}_{(t)} \in \mathbb{R}^{I^{(x)} \times I^{(y)}}$ as the pixelated representation of the image. $I^{(x)}$ is the width of the image, and $I^{(y)}$ is the height of the image.

2.1 Optical Flow

The OF is a vector field $\mathbf{v}(x,y,t)$ that models the velocity of the pixel at the position (x,y) [14]. It is derived by solving a PDE-constrained optimization problem:

$$\mathbf{v}(x,y,t) = \text{argmin}_{\mathbf{v}} ||\nabla I(x,y,t) \cdot \mathbf{v}(x,y,t) + I_t(x,y,t)||, \tag{1}$$

where $\nabla I(x,y,t)$ is the gradient vector of $I(x,y,t)$ with respect to its spatial coordinates, and $I_t(x,y,t)$ is the partial derivative of I with respect to its temporal coordinate.

The OF is computed numerically using the Lucas–Kanade method [15]. In this method, for every pixel with spatial coordinate (x,y), $v(x,y)$ is computed by solving the following optimization problem:

$$\mathbf{v}(x,y,t) = \text{argmin}_{\mathbf{v}} ||A\mathbf{v}(x,y) + b||^2, \tag{2}$$

where $A \in \mathbb{R}^{p \times 2}$ is the matrix consisting of finite-difference approximations of $\nabla I(x,y,t)$ at p neighboring pixels, and $b \in \mathbb{R}^p$ is the vector consisting of finite-difference approximations of $I_t(x,y,t)$. This is solved using the least squares method. In this paper, we will refer to $\mathbf{v}_{(t)} \in \mathbb{R}^{2 \times I^{(x)} \times I^{(y)}}$ as the pixel representation of the motion vector at time t.

2.2 Deep Q Learning

Deep Q Learning is a RL algorithm that involves training a DQN parameterized by weights w [1]. A DQN takes a state and action as input and outputs the following expected cumulative reward:

$$Q_w(s,a) = E\left[\sum_t \gamma^t r_t | s_0 = s, a_0 = a\right], \tag{3}$$

where r_t, s_t, a_t are the rewards, states, and actions at time step t. At each time step, a transition $\{s_t, a_t, r_t, s_{t+1}\}$ is recorded and used later to update the DQN using the Bellman equation:

$$w \leftarrow w - \alpha \nabla_w (r_t + \gamma \max_{a'} Q_w(s_{t+1}, a') - Q_w(s_t, a_t))^2, \tag{4}$$

where α is a learning rate. The weights of $Q_w(s', a')$ are frozen here. In our paper, the states will be images at time t, $\mathbf{I}_{(t)}$.

3 Deep Q Learning with Optical Flow embeddings

In this work, we will explore the intersection of RL with OF. OF gives information about the motion of sequential image data, which gives it importance in training RL models.

For an RGB frame at time step t, we preprocess it according to the evaluation protocol from [16]. We grayscale the frame and resize it to 84×84 to obtain $\mathbf{I}_{(t)}$

For $\mathbf{I}_{(t)}$, we compute the OF $\mathbf{v}_{(t)}$ using the Lucas–Kanade method. For compatibility with DQN, we scaled the OF by: $\mathbf{v}_{(t)} \leftarrow \texttt{np.uint8}(\texttt{np.clip}(8\mathbf{v}_{(t)} + 128, 0, 255))$. We then concatenate $\mathbf{I}_{(t)}$ with $\mathbf{v}_{(t)}$ to obtain the state $\mathbf{s}_t \in \mathbb{R}^{3\times 84\times 84}$, which we then pass into our DQN as input. We used the DQN implementation by [17] for this paper.

To assess the practicality of OF in Atari environment, we plotted $\mathbf{v}_{(t)}$ for an arbitrary time step in the Breakout-v5 and SpaceInvaders-v5 environment in figures 1 and 2. Based on the OF for the Breakout-v5 environment, it is clear that the ball is moving towards the bottom-left direction, and the platform is moving to the right. And the OF for the SpaceInvaders-v5 environment is fuzzy in comparison, but it clear that the bullet is moving upwards, and the user-controlled spaceship is moving to the right.

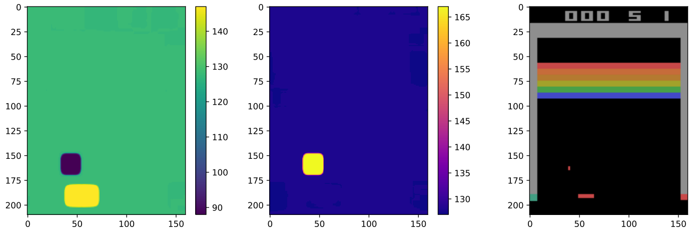

Fig. 1: Optical flow mapping of the motion in the x-direction (left), the motion in the y-direction (middle), of a time step in the Breakout environment (right)

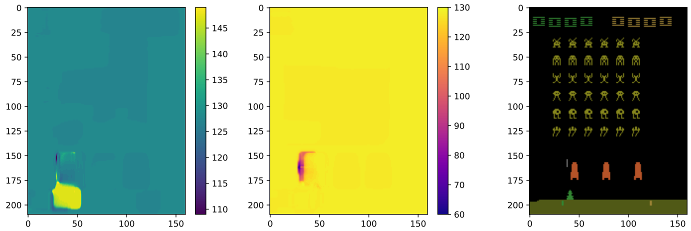

Fig. 2: Optical flow mapping of the motion in the x-direction (left), the motion in the y-direction (middle), of a time step in the SpaceInvaders environment (right)

We then proceed to test the DQN with the preprocessed data on 18 different Atari environments to see if OF helps with the training of these models.

4 Numerical Results

In this section, we will evaluate the benefit of OF in training a DQN models. To assess this, we will compare it to an algorithm where we concatenate the current frame $\mathbf{I}_{(t)}$ with the previous frame $\mathbf{I}_{(t-1)}$. At time step 0, we let $\mathbf{I}_{(-1)} = \mathbf{I}_{(0)}$. $\mathbf{I}_{(t)}$ is processed similarly as mentioned in section 3.

We train the DQN model on 18 different Atari environments for a million time steps each. To speed up the training process, we trained each model on the Cedar cluster in Compute Canada using 4 P100-12gb GPUs. Training each model took approximately 2 hours.

For a comparison of the two algorithms, we posted the learning curves for each environment in figure 3. In 8 out of the 18 environments, DQN with OF shows significantly higher rewards compared to DQN with concatenation. DQN with concatenation shows significantly higher rewards only in the Pooyan-v5 environment. In the remaining 9 environments, both algorithms show equal performance.

This method shows an overall improvement compared to a DQN that takes the concatenation of a frame with its previous frame as input. The reason DQN with OF shows improved performance is because the OF is a visual representation of the motion in an environment that no longer needs to be learned by the DQN to choose optimal actions. Thus, the results show that the combination of image data with motion representation can improve the performance of RL algorithms in vision-based environments such as Atari because it provides a richer and more informative representation for RL models.

5 Conclusion

In this paper, we present a study of the effectiveness of integrating optical flow information alongside image data and assess its impact on the performance of RL agents. We extract the OF from Atari environments on a frame-by-frame basis and concatenate it with the frame to as an input for our DQN. We demonstrate that this combination results in improved learning and higher rewards compared to a DQN that takes the concatenation of a frame with its previous frame as input.

This research contributes to the broader understanding of how visual information and motion cues can be leveraged to optimize RL in Atari games, ultimately paving the way for more capable and efficient agents in dynamic real-world scenarios. For future work, we would like to see how OF enhances the performance of RL algorithms outside the scope of video games. Applications include robotics such as mentioned in [5]. It will be interesting to see how the integration of OF in RL algorithms helps controllers deal with issues and uncertainty arising from integrating RL with experiments conducted in real life.

References

[1] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[2] R. S. Sutton, "Dyna, an integrated architecture for learning, planning, and reacting," *ACM Sigart Bulletin*, vol. 2, no. 4, pp. 160–163, 1991.

[3] R. Coulom, "Efficient selectivity and backup operators in monte-carlo tree search," in *International conference on computers and games*. Springer, 2006, pp. 72–83.

[4] D. Silver, A. Huang, and C. e. a. Maddison, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, p. 484–489, 2016.

[5] A. V. Nair, V. Pong, M. Dalal, S. Bahl, S. Lin, and S. Levine, "Visual reinforcement learning with imagined goals," *Advances in neural information processing systems*, vol. 31, 2018.

[6] L. Kaiser, M. Babaeizadeh, P. Milos, B. Osinski, R. H. Campbell, K. Czechowski, D. Erhan, C. Finn, P. Kozakowski, S. Levine *et al.*, "Model-based reinforcement learning for atari," *arXiv preprint arXiv:1903.00374*, 2019.

[7] A. Akutsu, Y. Tonomura, H. Hashimoto, and Y. Ohba, "Video indexing using motion vectors," in *Visual Communications and Image Processing'92*, vol. 1818. SPIE, 1992, pp. 1522–1530.

[8] R. Li, B. Zeng, and M. L. Liou, "A new three-step search algorithm for block motion estimation," *IEEE transactions on circuits and systems for video technology*, vol. 4, no. 4, pp. 438–442, 1994.

[9] C.-L. B. Lin and M.-C. Lee, "Efficient motion vector coding for video compression," US Patent 6983018, January 2006.

[10] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial intelligence*, vol. 17, no. 1-3, pp. 185–203, 1981.

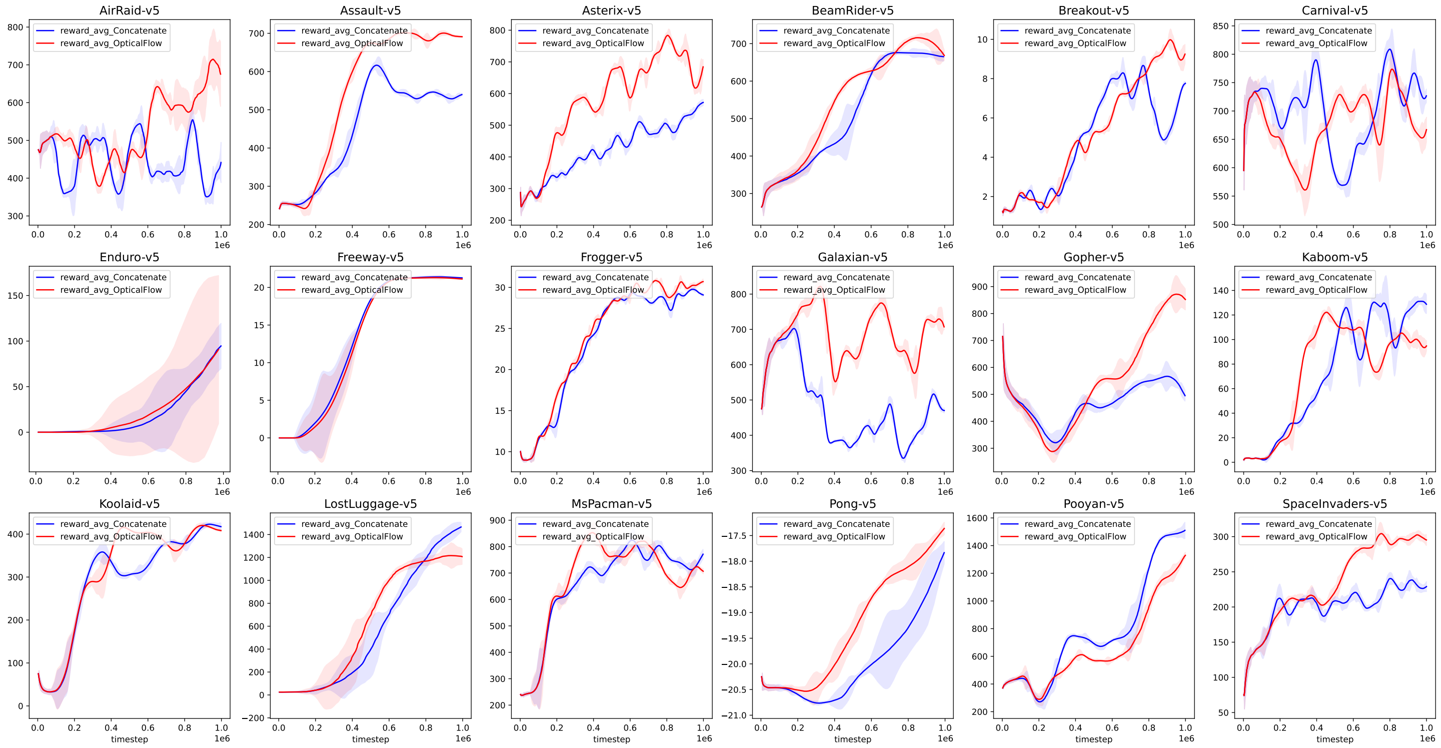

Fig. 3: Comparison of learning curves for Optical Flow (Red) compared to Concatenation (Blue)

[11] X. Zhuang, X. Zhou, M. A. Hasegawa-Johnson, and T. S. Huang, “Real-world acoustic event detection,” *Pattern recognition letters*, vol. 31, no. 12, pp. 1543–1551, 2010.

[12] W. Liu, S. M. Ayyoubzadeh, Y. Yu, I. Kezele, Y. Wang, X. Wu, and J. Tang, “Method, apparatus and system for adaptating a machine learning model for optical flow map prediction,” U.S. Patent 20230148384A1, May 2023.

[13] W. Liu, Y. Yu, Y. Wang, J. Lu, X. Wu, and J. Tang, “Method, device, and medium for generating super-resolution video,” U.S. Patent 20230148384A1, October 2023.

[14] M. Hasegawa-Johnson, “Lecture 6: Optical flow,” ECE 417: Multimedia Signal Processing, University of Illinois, 2021.

[15] B. D. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” in *IJCAI’81: 7th international joint conference on Artificial intelligence*, vol. 2, 1981, pp. 674–679.

[16] M. C. Machado, M. G. Bellemare, E. Talvitie, J. Veness, M. Hausknecht, and M. Bowling, “Revisiting the arcade learning environment: Evaluation protocols and open problems for general agents,” *Journal of Artificial Intelligence Research*, vol. 61, pp. 523–562, 2018.

[17] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, “Stable-baselines3: Reliable reinforcement learning implementations,” *Journal of Machine Learning Research*, vol. 22, no. 268, pp. 1–8, 2021.