IEOR E8100 Project Report: Image Registration using Deep Reinforcement Learning

Hengda He (hh2699), Jinxin Xiong (jx2383),
Nuntanut Raksasri (nr2747), Yice Luo (yl4243)

Abstract

Image registration is a process of aligning two or more images into the same coordinate with many applications in computer vision and medical image processing. Based on registration models, image registration can be divided into linear registration working with rigid body or affine transformations, and non-linear registration working with deformation fields. The variability/difference and correspondence between a pair of images can be quantified by such an affine transformation or a deformation field. Traditional image registration seeks to use optimization methods to estimate the optimal transformation or deformation, that minimizes the difference between a pair of images when transformation or deformation applied to one of them. But these methods are sensitive to local minimum, and the investigations of reinforcement algorithms in image registration are limited especially in non-linear registration. In this project, we propose a deep reinforcement learning algorithm to solve both linear and non-linear image registration problems. In a rigid body transformation experiment, we show that our proposed method outperforms a widely-used traditional method in the registration of image pairs with large variabilities/transformations. We also claim that our method is more interpretable, as we can visualize the optimal registration strategy. In another experiment with a non-linear registration problem, we propose the first reinforcement learning algorithm to cope with pixel-wise displacement, which is challenging as pixel-wise displacement requires a strong smoothness and diffeomorphism constraint. Here, we train the agent in a parameterized space, pixel-wise initial momentum, which can indirectly manipulate the pixel-wise displacement. As the state and action space is large, a reward and action map convolution are used to propagate rewards and actions across neighboring pixels. To better extract correspondences between image pairs, a correlation layer is proposed to use in our neural network, which improves the performance.

1 Introduction

Image registration aims to geometrically align two or more images into the same coordinate, and has been widely used in computer vision, medical images and self-driving cars [1, 2]. The goal of an image registration is to find the optimal transformation that best aligns the structure of interest in a moving image to a fixed image. The resulting optimal transformation gives a mapping from each pixel/voxel position in one image to a corresponding position in the other images. This method is a crucial step for image analysis in which valuable information is integrated from multiple images; i.e., images acquired at different time/sessions, from distinct viewpoints [3] or by different sensors, or from different imaging modalities [4]. Therefore, accurate registration of two or more images into the same coordinate system is important for integration (or fusion) of useful information from the images. Image registration involves an optimization problem to determine a linear transformation (rigid or affine) or non-linear deformation field $\phi^* \in T$ that minimizes the image matching

dissimilarity under certain regularization

$$\phi^* = arg \min_{\phi \in T} D(I_{Fixed}, \phi(I_{Moving})) + Regularization(\phi)$$

where ϕ is a transformation or deformation field based on the registration model, T is a set of possible transformations/deformation fields, I_{Moving} and I_{Fixed} denote the moving and the fixed image, respectively, $\phi(I_{Moving})$ is the moving image transformed/warped using the transformation/deformation ϕ, D is a dissimilarity (or similarity) metric measuring the alignment, *Regularization* term is usually used in non-linear deformation to enforce pre-defined desired properties into the deformation field such as smoothness or diffeomorphism.

Traditional methods try to compute the dense mapping between two images by minimizing an objective function with regard to some similarity criterion as described above. However, suffering from local minimum, many approaches have difficulties in handling large transformations/deformations or large variabilities in appearance. In our project, we propose to implement a robust RL algorithm that is able to work well in the large transformations/deformations cases. Recently, results using deep representation learning have been presented for learning similarity metrics [5], predicting the optical flow [6] or the large deformation diffeomorphic metric mapping-momentum [7]. These approaches either only partially remove the above-mentioned limitations as they stick to an energy minimization framework or rely on a large number of training samples derived from existing registration results. In [8], a reformulation of the non-rigid registration problem following a similar methodology, as in 3-D rigid registration of [9], was proposed. In order to optimize the parameters of a deformation model they apply an artificial agent (only learned from experience) that does not require explicitly designed similarity measures, regularization and optimization strategy. This proposed method overcomes limitations of traditional algorithms by learning optimal features for decision-making. Therefore, segmentation or handcrafted features are not required for the registration during testing. Additionally, a novel ground-truth generator was proposed to learn from synthetically deformed and inter-subject image pairs. However, the author was concerned that the deformation parametrization in the paper needs to be further evaluated. To solve that, in this project, we propose to use a better deformation parameterization from LDDMM [10], which is capable of working with large deformations compared to [8]. Rigid registration as in [9] could be included in the network or applied as preprocessing to improve results as shown in the experiments. Besides, the extension to multi-modal registration is desirable as proposed in [11] and [12]. In [11], a novel depth-CT registration method based on deep reinforcement learning (RL) was proposed. Their approach investigated the correlations between surface readings from depth sensors and internal body structures captured by the CT imaging. The experimental results demonstrated that their approach reached the best accuracy with the least deviation. The better performance compared to two original deep RL methods suggests that their modifications improved the network learning for the multimodal registration. Higher errors in the abdomen cases, compared to the chest cases, may be caused by the larger appearance variations. The proposed approach also had no limitations to be applied to register images from other modalities. To further improve performance, combining the surface metric together with the contextual information and extra efforts are also required to improve the training and testing efficiency. In [12], a new learning paradigm in the context of image registration based on RL was presented. Different from the previous work [11] that also build upon RL, their method devised a combined policy network and value network to respectively generate the action probability and state value, without the need of extra storage on exploration history. The network was trained using A3C algorithm, where a customized reward function was also leveraged to encourage robust image registration. This trained network is further incorporated with a lookahead inference to improve the registration capability. In [13], a multi-

agent system with an auto attention mechanism for robust and efficient 2D/3D image registration was proposed. To reduce the number of degrees of freedom that need to be sampled in training, an individual agent is trained with dilated Fully Convolutional Network to perform registration in MDP by observing a local region. The final action is then taken based on the proposals from multiple agents and weighted by their corresponding confidence levels. This method is able to achieve significantly higher robustness than the state-of-the-art 2D/3D registration methods.

Inspired by [14] and the further improvement of [12], [13] and [8], we propose a RL algorithm with pixel-wise rewards (pixelRL) for image registration using the A2C algorithm. As in these literature, deep RL has been achieving great performance in image registrations in recent years. But, the applications of deep RL for image registration have not been thoroughly investigated, especially in non-linear registration. In experiment 1, we implement a RL algorithm to solve a rigid body image registration problem with potential applications in multimodal image registration, and have demonstrated that our method outperforms the traditional method in large transformation cases, where traditional method is stuck in a local minimum. In experiment 2, we implement a RL algorithm to work with pixel-wise displacement deformation fields for non-linear image registration, which has not been investigated before. In our method, we use a reward map convolution, as well as a novel action map convolution to propagate rewards and actions across neighboring pixels, which improves the training performance. Compared to other studies [13, 12, 8, 15] using a concatenation layer, we use a correlation layer [6], that performs multiplicative patch comparisons between two feature maps to extract correspondence, which improves the training performance. Moreover, the proposed method can enhance the explainability and interpretability as the registration strategy can be visualized.

2 Methods

2.1 Rigid body transformation

In this experiment, we applied RL to determine a sequence of translations and rotations to align the perturbed/moving image to the target image. Also, we will show that the results by using RL, will be more robust and will outperform the supervised learning methods to learn the transformation parameters. Denote the target fixed image as $\mathbf{F}$, and the initial moving image/perturbed image as $\mathbf{M_0}$. Then, our problem is that we want to find a sequence of transformation parameters $\mathbf{p_t}$ to make the transformed moving image being as close as possible to the fixed target image. That is the distance between $\mathbf{F}$ should be close enough to $\mathbf{M_t} = \mathbf{p_t} \circ \mathbf{M_0}$. We consider the action space to be of size 6: $+1, -1$ pixel along x, $+1, -1$ pixel along y, $+30, -30$ degrees. And then, the transformation parameters are updated by $\mathbf{p_t} = \mathbf{p_{t-1}} + \mathbf{a_t}$. To make sure that the agent will not move the number outside of the frame and help the agent to learn the most efficient strategy, we design the reward function as following: $\mathbf{R}(\mathbf{F}, \mathbf{M_t}, \mathbf{p_t})$:

$$\mathbf{r_t} = \mathbf{R}(\mathbf{F}, \mathbf{M_t}, \mathbf{p_t}) = \begin{cases} -R_1, \text{if moving out of the frame} \\ -R_1, \text{if rotate more than half circle} \\ +R_2, \text{if } ||\mathbf{F} - \mathbf{M_t}||_2^2 < threshold \\ -||\mathbf{F} - \mathbf{M_t}||_2^2, \text{otherwise} \end{cases}$$

We used actor-critic method to train the agent, the only difference between our model and the lab4 is that we used the concatenation of the fixed and moving image as the input and let the actor and critic have the same feature extractor. To make the model less sensitive to the outliers, instead

of using the mean-squared error for the critic, we used huber loss. More details about the MDP formulation and the model can be found in the Appendix A.

2.2 Non-linear deformation

In experiment 2, we have a similar MDP setup as in experiment 1, but here we aim to implement the algorithm to work with more complicated non-linear image registration models. The implementation of RL algorithm in image registration with affine transformations has been well investigated in recent years [13, 16, 12, 9], but to the best of our knowledge, there is only one study of implementing RL algorithm in non-linear image registration [8]. Compared to affine transformations with degrees of freedom equal to or less than 12, implementing RL to solve non-linear image registration problem is challenging since the degrees of freedom is equal to $H \times W \times 2$ in the 2D case. In [8], the authors proposed to reduce the degrees of freedom by parameterizing the deformation fields with parameters less than a hundred, and proposed a supervised learning method to learn these parameters directly. But the deformation model in [8] is not capable of working with large deformations due to the deformation field dimension reduction and B-spline model used in the study. The ground truth is usually hard to obtain in image registration, and the authors proposed to use ground truth estimated from other algorithms, which could limit their RL performance. Here, we propose to incorporate pixelRL [14] and work with pixel-wise displacement fields. To work with large deformations, we used LDDMM deformation model (more details in Appendix B) and parametrized the deformation by initial parameters $\mathbf{m_0}$, which is with shape $(H, W, 2)$. And we trained the agent in an unsupervised way.

Instead of directly cope with the displacement ϕ, which requires a strong regularization such as smoothness or diffeomorphism, our agent works in the space of initial momentum $\mathbf{m_0}$. The agent is trained to learn a sequence of actions to update the displacement $\phi_{m_0^t}$, such that it can bring the initial moving image M_0 as close as possible to the target fixed image F. Our state is defined as a pair of fixed image F and moving image M_t at current time step t. Under LDDMM settings, M_t is obtained by $\mathbf{M_t} = \phi_{\mathbf{m_0^t}} \circ \mathbf{M_0}$. We consider the action space to be of size 5 for each pixel $(H, W, 5)$: $+\delta, -\delta$ changes of the initial momentum along x or y, and no change. And then, the transformation parameters are updated by $\mathbf{m_0^{t+1}} = \mathbf{a_t} \circ \mathbf{m_0^t}$. The reward $\mathbf{r_t}$ is defined as $\mathbf{R}(\mathbf{F}, \mathbf{M_t}, \mathbf{m_0^t})$ (see more details in Appendix B). We used actor-critic method to train the agent, which has the same feature extractor for fixed image and moving image. Instead of directly concatenating features from the fixed and moving images as in [13, 12, 8, 15], we use a correlation layer [6] to extract correspondences between the feature maps, which performs multiplicative patch comparisons between two feature maps. The architecture can be found in the Appendix B. The design of the reward function will give reward for the agent if the agent improves the matching of the image pairs with the action at that step, and will penalize the agent if the action will move the image pairs away to each other at that step. With the same idea in [14], since the pixel-wise state and action space is too large and the reward could be very sparse, we want to propagate any rewards from the pixels to its neighbors. Thus, the reward and action map is convolved with a Gaussian kernel.

3 Experiments and Results

3.1 Rigid body transformation

We use tf.keras.preprocessing.image.apply_affine_transform to perform the transformations and generate the data. As any other affine transformation packages, the function will do center offset and intensity interpolation apart from the intended transformations. For this reason, even if we know

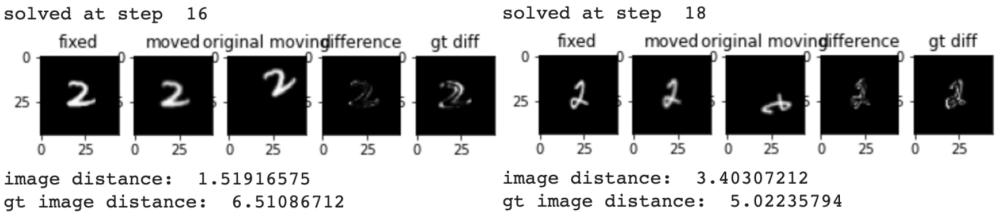

Figure 1: Results of Experiment 1. In most cases the agent can successfully find parameters to align the moving images to the fixed images, and even overcome the interpolation inaccuracy during transformation. And the distance between the result and the target image can be smaller than that obtained by the supervised learning method or even the ground truth (due to interpolation inaccuracy)

the ground truth parameters and apply the inverse transformation on the moving image, we will not be guaranteed to get a result that is close enough to the fixed image. So, if we learn the transformation parameters in a supervised way, we may always learn a sub-optimal solution. We resize the the MNIST dataset to 44 by 44 and the perturbed/moving image is translated within +/- 15 pixels along x and y axis and rotation with +/- 6×30 degrees. We set the threshold in the reward function to be 3.5, steps per episode to be 55, and the problem is said to be solved if the running average reward is larger than 250. As shown in **Figure:** 1, in most of the cases the agent can successfully find parameters to align the moving images to the fixed images. And the distance between the result and the target image can be smaller than the result obtained by the supervised learning method or even the ground truth. Here is a step by step video for the entire process of the registration. We can see that the agent will first translate the number to the center and to be as close to the target image as possible and then do the rotation, which should be the optimal strategy in this case. As discussed in the introduction section, traditional image registration methods have difficulties in the tasks requiring large transformations. To evaluate the robustness of our algorithm in coping with image samples with large variability, we compared our method with a widely-used traditional image registration method (*imregister* function from Matlab) in simulated unseen image registration samples with large perturbations/transformations. Results are shown in **Figure:** 2.

3.2 Non-linear deformation

We use LDDMM to perform the deformation. The images are scaled into the range between 0 and 1, we set the threshold in the reward function as 85, steps per episode as 20. As shown in **Figure:** 4 of Appendix B, the agent was able to register the images very well at the bottom parts of the digit but still have some mismatch at the top structures. Here is a step by step video.

4 Conclusion and Discussion

In this project, we investigated the implementation of RL algorithms in solving image registration problems. Traditional image registrations are typically based on an optimization algorithm, and suffer from the local minimum problem especially in applications requiring large transformations and deformations. Through literature search, we implemented our own RL algorithm to solve the rigid body transformation image registration problem as in experiment 1. And we have demonstrated in our results that our proposed RL algorithm performs better in large transformation cases compared to a traditional image registration method from Matlab. The implementation of RL in

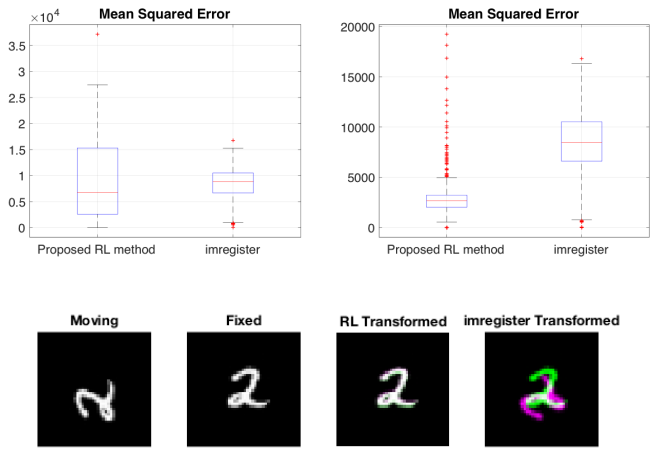

Figure 2: Compare our proposed RL method with a traditional image registration method *imregister* from Matlab. Top Left: 500 registration sample pairs simulated with rotation within {-120, -90, 90, 120}, translation range from -12 to +12. Top Right: 500 registration sample pairs simulated with rotation within {-120, -90}, translation range from -12 to +12. Bottom: Our proposed method align two images very well in large transformations, whereas traditional method is stuck in a local minimum. In the transformed images, green denotes the fixed image, pink denotes the transformed image, and white denotes overlap between the fixed and transformed images.

non-linear image registration has not been investigated well in the field with only one publication [8] in recent years. In our project, we propose the first RL algorithm in non-linear image registration that works with pixel-wise deformation. Due to the fact that the pixel-wise displacement requires a strong regularization, directly applying the action map to pixel-wise displacement is difficult. Here, we propose to use RL algorithm with LDDMM image registration model, where the pixel-wise displacement is parameterized by a pixel-wise initial momentum, and the agent works in the space of initial momentum instead of the displacement directly. This parameterization allows the topology of the images in the state preserved during the action application at each step, thus making an indirect manipulation of the pixel-wise deformation through RL actions possible. In our implementation, we used a reward map and action map convolution to propagate rewards and actions across neighboring pixels, and we also incorporated a correlation layer into our model to extract correspondence between feature maps. Moreover, the proposed method can enhance the explainability and interpretability as the registration strategy can be visualized.

In our experiments, we use small size and clean dataset, MNIST, as the training and testing dataset, with the idea of focusing more on the implementation of RL and the concern of computational limitations. In real world application, as working with a rigid-body transformation model, our implementation in experiment 1 can be extended to work with applications such as multimodal medical image registration, motion correction for image time frames, and homography registration. As for experiment 2, which works with non-linear deformations, our implementations can be extended to applications such as image geometric distortion correction, studying brain atrophy, or spatial normalization in funtional MRI. As for RL algorithms, we basically use a vanilla actor-critic algorithm. However, there are many advanced technique and algorithms we should try, like A3C and PPO. It's a good direction to do experiments in the future. Transfer learning has been widely used in computer vision, such as using pre-trained network like VGG or ResNet as a feature extractor for more complicated applications. This technique can also be applied in our algorithm.

References

[1] Francisco PM Oliveira and Joao Manuel RS Tavares. "Medical image registration: a review". In: *Computer methods in biomechanics and biomedical engineering* 17.2 (2014), pp. 73–93.

[2] Vijay Rengarajan, Yogesh Balaji, and AN Rajagopalan. "Unrolling the shutter: Cnn to correct motion distortions". In: *Proceedings of the IEEE Conference on computer Vision and Pattern Recognition*. 2017, pp. 2291–2299.

[3] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. "Deep image homography estimation". In: *arXiv preprint arXiv:1606.03798* (2016).

[4] Tongxue Zhou, Su Ruan, and Stéphane Canu. "A review: Deep learning for medical image segmentation using multi-modality fusion". In: *Array* 3 (2019), p. 100004.

[5] Martin Simonovsky et al. *A Deep Metric for Multimodal Registration*. 2016. arXiv: 1609.05396 [cs.CV].

[6] Philipp Fischer et al. "Flownet: Learning optical flow with convolutional networks". In: *arXiv preprint arXiv:1504.06852* (2015).

[7] Xiao Yang, Roland Kwitt, and Marc Niethammer. "Fast Predictive Image Registration". In: Oct. 2016, pp. 48–57. ISBN: 978-3-319-46975-1. DOI: 10.1007/978-3-319-46976-8_6.

[8] Julian Krebs et al. "Robust Non-rigid Registration Through Agent-Based Action Learning". In: Sept. 2017, pp. 344–352. ISBN: 978-3-319-66181-0. DOI: 10.1007/978-3-319-66182-7_40.

[9] Rui Liao et al. *An Artificial Agent for Robust Image Registration*. 2016. arXiv: 1611.10336 [cs.CV].

[10] John Ashburner and Karl J Friston. "Diffeomorphic registration using geodesic shooting and Gauss–Newton optimisation". In: *NeuroImage* 55.3 (2011), pp. 954–967.

[11] Kai Ma et al. "Multimodal Image Registration with Deep Context Reinforcement Learning". In: Sept. 2017, pp. 240–248. ISBN: 978-3-319-66181-0. DOI: 10.1007/978-3-319-66182-7_28.

[12] Shanhui Sun et al. "Robust Multimodal Image Registration Using Deep Recurrent Reinforcement Learning". In: *Computer Vision – ACCV 2018*. Ed. by C. V. Jawahar et al. Cham: Springer International Publishing, 2019, pp. 511–526. ISBN: 978-3-030-20890-5.

[13] Shun Miao et al. *Dilated FCN for Multi-Agent 2D/3D Medical Image Registration*. 2017. arXiv: 1712.01651 [cs.CV].

[14] R. Furuta, N. Inoue, and T. Yamasaki. "PixelRL: Fully Convolutional Network With Reinforcement Learning for Image Processing". In: *IEEE Transactions on Multimedia* 22.7 (2020), pp. 1704–1719. DOI: 10.1109/TMM.2019.2960636.

[15] Xiao Yang et al. "Quicksilver: Fast predictive image registration–a deep learning approach". In: *NeuroImage* 158 (2017), pp. 378–396.

[16] Shanhui Sun et al. "Robust Multimodal Image Registration Using Deep Recurrent Reinforcement Learning". In: *Lecture Notes in Computer Science* (2019), pp. 511–526. ISSN: 1611-3349. DOI: 10.1007/978-3-030-20890-5_33. URL: http://dx.doi.org/10.1007/978-3-030-20890-5_33.

Appendix A (Experiment 1)

A.1 Rigid body transformation

Translation along x by t_x, y by t_y:

$$\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

Rotation by θ:

$$\begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Scale along x by s_x, along y by s_y:

$$\begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

In this report, we just consider translation and rotation. Therefore, for each point/pixel, the transformation is given by:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & t_x \\ \sin(\theta) & \cos(\theta) & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

To generate the data, we use tf.keras.preprocessing.image.apply_affine_transform, which will return the transformed image given the parameters. When doing transformation, the function will offset the center. Assume the width of the image is W and the height is H, take $o_x = W/2 - 0.5$, $o_y = H/2 - 0.5$. Denote the previous transform matrix is T,

$$T' = \begin{bmatrix} 1 & 0 & o_x \\ 0 & 1 & o_y \\ 0 & 0 & 1 \end{bmatrix} \cdot T \cdot \begin{bmatrix} 1 & 0 & -o_x \\ 0 & 1 & -o_y \\ 0 & 0 & 1 \end{bmatrix}$$

Before returning the transformed image, the function will do some interpolation. For simplicity, we can regard the transformation function as a function, which will do the intended transformation plus an unknown noise. For this, even if we know the ground truth parameters for the perturbed image, using the function to do the inverse transformation may not guaranteed to return the original image.

In this report, we trained a agent to learn how to transform a perturbed moving image to be as close as possible to the fixed target image.

A.2 MDP Formulation

A.2.1 State

Denote:

- Fixed image: $\mathbf{F}$
- Initial moving image: $\mathbf{M_0}$

- Initial parameters $\mathbf{p_0} = [0.0, 0.0, 0.0]$, all initialize to be zero
- Parameters at time t: $\mathbf{p}_t = [\theta_t, x_t, y_t]$
- Transformed moving image at time t: $\mathbf{M_t} = \mathbf{p_t} \circ \mathbf{M_0}$
- State at time t: the concatenation of the fixed and current moving image, that is $\mathbf{S_t} = [\mathbf{F}; \mathbf{M_t}]$

A.2.2 Action

In this simple problem setting, we consider the action space to be size 6: $+1, -1$ pixel along x, $+1, -1$ pixel along y, $+30, -30$ degrees. Denote the chosen action at time t as $\mathbf{a_t}$, it will be one of $[+30.0, 0.0, 0.0], [-30.0, 0.0, 0.0]$,
$[0.0, +1.0, 0.0], [0.0, -1.0, 0.0], [0.0, 0.0, +1.0], [0.0, 0.0, -1.0]$.

Once we obtained $\mathbf{a_t}$, we can update the transform parameters by $\mathbf{p_t} = \mathbf{p_{t-1}} + \mathbf{a_t}$.

A.2.3 Rewards

The rewards is defined as the negative of distance between the fixed image $\mathbf{F}$ and the current moving image $\mathbf{M_t}$. To help the agent to learn not to move out of the frame, we terminate and set the reward to a relative large negative value $(-R_1)$. Similarly, to help the agent to learn an efficient strategy, we terminate and set the reward to a large negative value if the parameter trying to rotate more than half circle. The problem is solved if the distance between the fixed and moving images is less than a *threshold*. R_1, R_2 can be set as any large number relative to the average total reward per episode.

A.3 Algorithm and Model

In this report, we use the Actor-Critic method to train the agent. Denote:

- Steps per episode: $\mathbf{T}$, may vary every episode
- Reward at time t: $\mathbf{r_t}$
- Expected return at time t: $\mathbf{G_t} = \sum_{t'=t}^{T} \gamma^{t'-t} \mathbf{r_{t'}}$, $\gamma \in (0, 1)$. To stabelize training, we standardize the returns to mean 0 and standard deviation 1
- Policy/ Actor parameterized by θ: π_θ
- Value function/ Critic parameterized by θ: V_θ^π

A.3.1 Model

We use a network to represent π_θ and V_θ^π. The network will take the current state, the concatenation of $\mathbf{F}, \mathbf{M_t}$ as the inputs, as the input, and pass the input to several convolution and maxpooling layers. Then the result are flattened and connected to fully connected layers which represent the actor, with size 6, and the critic, with size 1. A visualization of the network architecture in shown in Figure 3.

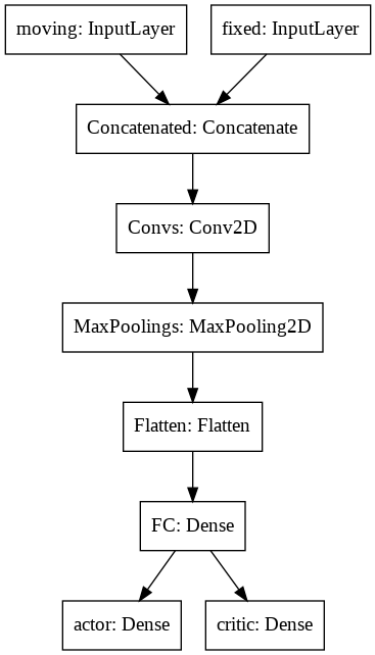

Figure 3: Model architecture

A.3.2 Algorithm

We have losses for actor and critic, denoted as L_{actor} and L_{critic}, respectively. And the total loss is defined as $L = L_{actor} + L_{critic}$.

$$
\begin{aligned}
L_{actor} &= -\sum_{t=1}^{T} \log \pi_\theta(a_t|s_t) \left[G_t - V_\theta^\pi(s_t)\right] \\
&= -\sum_{t=1}^{T} \log \pi_\theta(a_t|s_t) \left[\sum_{t'=t}^{T} \gamma^{t'-t} R(a_{t'}, s_{t'}) - V_\theta^\pi(s_t)\right] \\
L_{critic} &= L_\delta(G_t, V_\theta^\pi)
\end{aligned}
$$

where, L_δ is the Huber Loss, which is less sensitive to outliers in data than squared-error loss.

$$
L_\delta(y, f(x)) = \begin{cases} \dfrac{1}{2}(y - f(x))^2, |y - f(x)| \le \delta \\ \delta|y - f(x)| - \dfrac{1}{2}\delta^2, \text{otherwise} \end{cases}
$$

The entire procedure is shown in Algorithm: 1

Appendix B (Experiment 2)

B.1 LDDMM: Large deformation diffeomorphic metric mapping

In LDDMM [10], image warping is described as a time-varying flow quantified by a transport equation $d\phi(x,t)/dt = v(\phi(x,t),t)$. With time denoted by $t \in [0,1]$, spatial space by $x \in \Omega$, displacement vector field by $\phi(x,t)$, and velocity vector field by $v(x,t)$. This ordinary differential

Algorithm 1: Training procedure for the Actor-Critic Algorithm

Input: $F, M_0, P_0 = [0.0, 0.0, 0.0]$, initialize π_θ, V_θ^π networks
for $e = 1, ..., E$ **do**
  for $t = 1, ..., T$ **do**
    Input $[F; M_{t-1}]$ to the network and get $[a\_prob_t, v_t]$
    Sample $a_t \sim a\_prob_t$, update $p_t = p_{t-1} + a_t$
    Apply Transformation: $M_t = p_t \circ M_0$
    Collect the reward: $r_t = R(F, M_t, p_t)$
  end for
  Collected $\{a\_prob_t\}_{t=1}^T, \{v_t\}_{t=1}^T, \{r_t\}_{t=1}^T$
  Calculate expected return $\{G_t\}_{t=1}^T$
  Calculate the loss: $L = L_{actor} + L_{critic}$
  Updata parameters: $\theta \leftarrow \theta - \alpha \nabla_\theta L$
end for

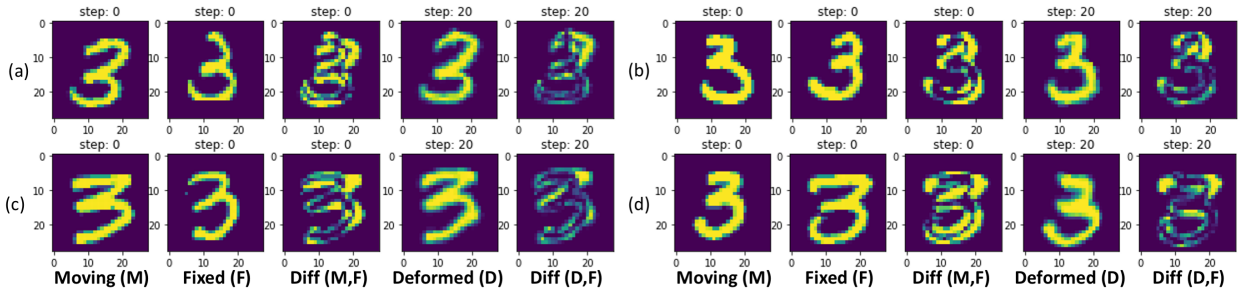

Figure 4: Results of Experiment 2. The agent was able to register the images very well at the bottom parts of the digit but still have some mismatch at the top structures

equation has the solution of ϕ with initial condition $\phi(x, 0) = x$. The final displacement vector field is taken as the end point of this image warping flow which is $\phi(x, 1)$. In the geodesic shooting setting, the image warping flow is parameterized via the initial momentum vector field m_0 and the initial condition $\phi(x, 0) = x$, from which the warping flow ϕ can be specified [15].

B.2 Algorithm and Model

The reward $\mathbf{r_t}$ is defined as $\mathbf{R}(\mathbf{F}, \mathbf{M_t}, \mathbf{m_0^t})$:

$$\mathbf{r_t} = \mathbf{R}(\mathbf{F}, \mathbf{M_t}, \mathbf{m_0^t}) = \begin{cases} -R_1, \text{if moving out of the frame} \\ +R_2, \text{if } MSE(\mathbf{F}, \mathbf{M_{t+1}}) < threshold \\ MSE(\mathbf{F}, \mathbf{M_t}) - MSE(\mathbf{F}, \mathbf{M_{t+1}}), \text{otherwise} \end{cases}$$

where $-R_1$ is a negative penalty, and $+R_2$ is a positive bonus for matching the images well enough. A visualization of the network architecture in shown in Figure 5.

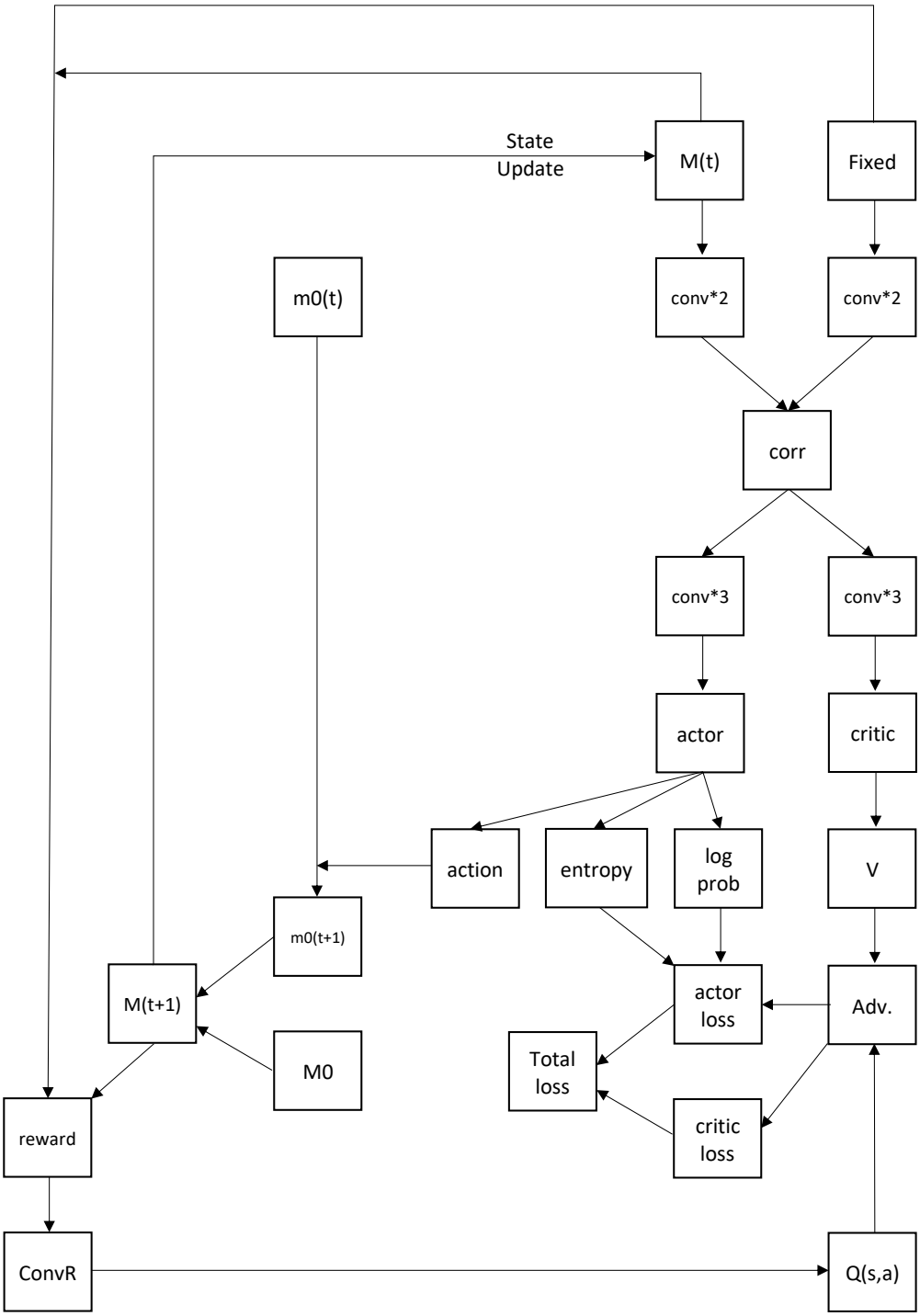

Figure 5: Model architecture in Experiment 2