

Stabilizing and Accelerating Autofocus with Expert Trajectory Regularized Deep Reinforcement Learning

Supplementary Material

6. Dataset

We also notice the issues as mentioned by Choi *et al.* [7] about using the multi-view stereo algorithm in [11] for data labeling. Specifically, there are labeling errors in regions with few textures or significant focus breathing. Additionally, labeling for this dataset requires the depth maps, which makes it inconvenient to extend to other datasets. Moreover, when reproducing the labeling process introduced by [16], we notice that the inverse depth range [0.2m, 100m] does not match with the focus distance range [0.102m, 3.910m]. This will result in half of the lens positions being unable to be labeled, since the lens positions indexed between 24 and 48 correspond to the focus distances ranging from 0.198m to 0.102m, which are below the lower bound of the inverse depth range. Therefore, the labeling method used in [16] is not adopted in this paper.

In contrast, Ho *et al.* [17] calculate the contrast for each region of interest (RoI), and use the lens position corresponding to the patch with the highest contrast in a focal stack as the in-focus position. This method is more convenient, since the contrast of any picture is easy to calculate. In addition, we believe that autofocus is essentially an image-based task rather than a depth-based one. In scenes with multiple DoFs and a large background with pure color, with depth-based labeling, choosing the median of depths in the corresponding stack to calculate the ground-truth (GT) index may result in focusing on the background instead of the target object, leading to the final image of defocused target objects.

Since the contrast-based methods [30, 40, 42] predict the in-focus position by maximizing the sharpness of the images and the focal stack of images has already been captured during the labeling stages, using the sharpness metric as the reference for labeling is convenient and proper. Thus, in this paper, we follow these contrast-based methods to get the label of the dataset. In particular, we adopt the six metrics, i.e., Laplacian [40], TenenGrad [42], Gradient Magnitude Variance [30], Census Transform [55], Normalized Cross-Correlation [3, 13], and Normalized SAD [13] to label the dataset. Among them, the first three are contrast-based metrics, usually with a higher accuracy, while the latter three are dual-pixel-based metrics, which can further utilize the information from dual-pixel data. We normalize every metric to the range of [0, 1] with the Min-Max Normalization. Since the contrast-based metrics demonstrate a higher accuracy, we assign the weight [0.25, 0.2, 0.2, 0.1, 0.15, 0.1] to the six normalized metrics to calculate the weighted av-

erage score. The lens position with the maximum weighted average score is then regarded as the in-focus position.

6.1. Contrast-Based Metrics

For a given focal stack $\{I_p^k\}$, the contrast-based approaches predict the in-focus position by searching for the focal position that can maximize the contrast metric ϕ of the patch I_k^p . Here, we introduce three typical metrics of this category. We let (x, y) denote the coordinates of an image pixel, Δ denote the operator on the image, and $\Delta[x, y]$ represent the response value at the pixel (x, y) .

Energy of Laplacian [40]. The input patch is convolved with a discrete Laplace operator, followed by squaring and summing the responses:

$$\phi = \sum_{x,y} \Delta[x, y]^2, \quad \Delta = I * \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}. \quad (11)$$

Mean Gradient Magnitude [42]. The input patch is convolved with two Sobel filters, followed by calculating the norm of the two responses. This approach is sometimes referred to as ‘‘TenenGrad’’:

$$\phi = \frac{1}{n} \sum_{x,y} \sqrt{\Delta_x[x, y]^2 + \Delta_y[x, y]^2},$$

$$\Delta_x = I * \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 1 & +1 \end{bmatrix}, \quad \Delta_y = I * \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix}. \quad (12)$$

Gradient Magnitude Variance [30]. This method calculates the variance of the gradient magnitudes in Eq. (12) instead of the norm:

$$\phi = \text{Var} \left(\sqrt{\Delta_x[x, y]^2 + \Delta_y[x, y]^2} \right). \quad (13)$$

6.2. Dual-Pixel-Based Metrics

Considering that the dataset consists of a series of dual-pixel patch pairs $\{(L, R)\}$, we adopt three phase-based approaches to fully utilize the phase information. For each focal stack $\{I_k^p\}$, we compute a disparity score ψ for each left-right patch pair (L, R) , and identify the focal index that maximizes this score as the in-focus index. To reduce the effect of images’ variation in global brightness, each patch

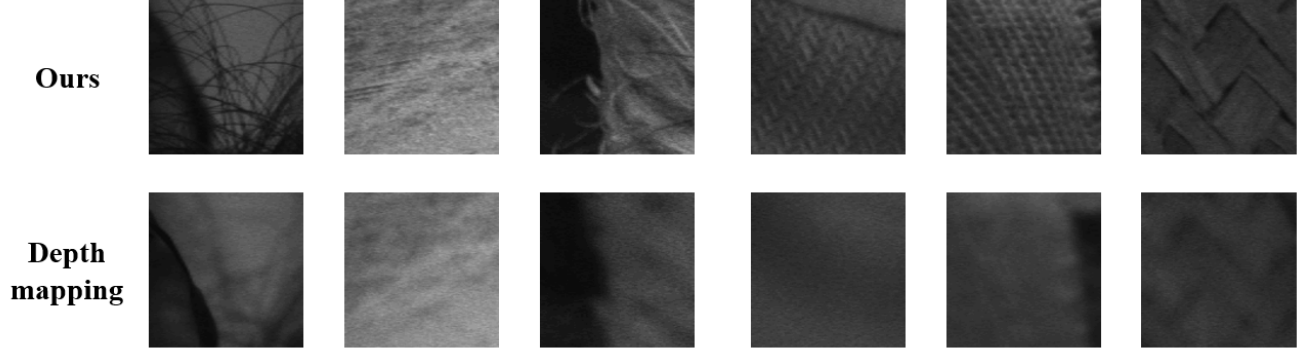


Figure 8. Visualization of the GT labeled by our methods and [16] (marked as "Depth mapping"). Ours are slightly sharper.

is normalized by subtracting its mean and dividing it by its standard deviation before computation.

$$\hat{L} = \frac{L - \mu(L)}{\text{Var}(L)}, \quad \hat{R} = \frac{R - \mu(R)}{\text{Var}(R)}. \quad (14)$$

Census Transform [55]. The score is calculated by summing the Hamming distance between the two census-transformed images. We adopt the negative score value as our final metric, ensuring that the position with the maximum score is the in-focus position. Let (x, y) denote the coordinates of an image pixel, $I[x, y]$ denote the pixel value at the patch I , and $\text{census}(I)[x, y]$ denote the census-transformed value of the pixel (x, y) at the patch.

$$\begin{aligned} \psi &= \sum_{x,y} \left| \text{census}(L)[x, y] - \text{census}(R)[x, y] \right|_0, \\ \text{census}(I)[x, y] &= \left[I[x + \Delta_x, y + \Delta_y] > I[x, y] \right] \quad (15) \\ & \quad \left[\Delta_x, \Delta_y \in [-1, 0, 1], \Delta_x \neq \Delta_y \neq 0 \right] \\ \psi_{final} &= -\psi. \end{aligned}$$

Normalized Cross-Correlation [3, 13]. This approach calculates the inner product of the normalized left and right patch pairs as the score:

$$\psi = \langle \hat{L}, \hat{R} \rangle. \quad (16)$$

Normalized SAD [13]. This approach computes the sum of absolute deviations between the normalized patch pairs. To make the in-focus patches those with the largest deviations, we take the negative score value.

$$\begin{aligned} \psi &= \sum_{x,y} |\hat{L}[x, y] - \hat{R}[x, y]| \\ \psi_{final} &= -\psi. \end{aligned} \quad (17)$$

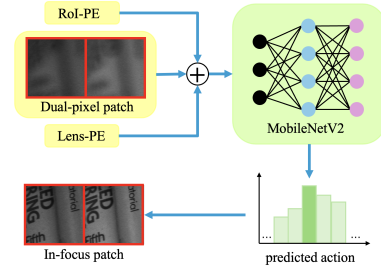


Figure 9. Inference procedure of the actor network.

7. Actor Network

To fully utilize the sequential multi-step information of the autofocus process, we use the ordinal regression loss [10] to pre-train the actor network, which is defined as:

$$y_k = \frac{e^{-(r_f - r_k)^2 / T}}{\sum_{i=1}^n e^{-(r_f - r_i)^2 / T}}, \quad (18)$$

where f is the actual in-focus position, y_k is a target probability distribution to model the probability of the model predicting each focal position, r_k represents the rank of focal position k , and T is a temperature parameter that controls the sharpness of the target distribution. Since the conversion from the absolute position label to the relative movement label is linear, r_k actually represents the rank of movement k at the current focal position in this work. Our goal is to make the output probability distribution of the model as close as possible to this target distribution. The inference procedure of the actor network is shown in Fig. 9.

8. Offline Expert Trajectory

In this paper, we expect to obtain an expert trajectory $\{g(\cdot), GT, (s_0, a_0), \dots, (s_n, a_n)\}$, where GT represents the in-focus position, (s_t, a_t) represents the state-action pair at time step t , and $g(\cdot)$ represents the transforming function from the lens position k_n to the state s_n in this trajectory, which is generated or refined by human knowl-

Algorithm 2 Expert trajectory generation 2.

Input: $\mathbb{S} = \{(s_0, g(\cdot), GT)\}_N$ and n ;

Output: \mathcal{D}^E ;

```

1:  $\mathcal{D}^E = \emptyset$ ;
2: for all  $(s_0, g(\cdot), GT)_i \in \mathbb{S}$  do
3:    $k_0 \leftarrow g_i^{-1}((s_0)_i)$ ;
4:    $a_0 \leftarrow GT_i - k_0$ ;
5:    $s_1 \leftarrow g_i(GT_i)$ ;
6:   for all  $j = 1, 2, \dots, n$  do
7:      $s_j \leftarrow g_i(GT_i)$ ;
8:      $a_j \leftarrow 0$ ;
9:   end for
10:   $\mathcal{D}^E \leftarrow \mathcal{D}^E \cup \{g(\cdot), GT, (s_0, a_0), \dots, (s_n, a_n)\}_i$ ;
11: end for
12: return  $\mathcal{D}^E$ .

```

edge. Ideally, an expert trajectory must satisfy the following conditions:

- The final position k_n must be within the DoF, i.e., $k_n \in \mathbf{k}^*$, to promise the sharpness, where \mathbf{k}^* denotes the set of lens positions within the DoF.
- $k_1 \leq k_2 \leq \dots \leq k_n$ or $k_1 \geq k_2 \geq \dots \geq k_n$, for no occurrence of focus hunting (FH).
- The lens movement distance across steps must gradually decrease, i.e., $|k_{i+1} - k_i| \leq |k_i - k_{i-1}|$, to avoid overshooting and get a smoother trajectory.

Since we mainly concentrate on the multi-step performance, moving the lens to the in-focus position in a single step may not be necessary. Meanwhile, if we set the trajectory to be correctly focused in single-step and the following 2– to 4–actions are 0 (stop), it is more likely to sample a zero action, which is harmful for exploration in RL. Besides Algorithm 1 in the main text, we design the other two algorithms, i.e., Algorithm 2 and Algorithm 3, to generate the expert trajectories and mix these trajectories in a single dataset.

Fig. 10-Fig. 12 visualize the generation or refinement process of Algorithm 1-Algorithm 3, respectively. In Algorithm 1, for a given trajectory, we use the in-focus position as the axis of symmetry and reflect all positions at the original trajectory into the interval range between the in-focus position and the initial position. Then, we re-sort these positions and consider the fixed trajectory as an expert trajectory. In Algorithm 2, for a given trajectory, we extract the initial position and in-focus position directly, and the in-focus position is appended after the initial position until reaching the length of the original trajectory. In Algorithm 3, for a given trajectory, we also extract the initial position and in-focus position directly, and calculate the distance d between the initial position and in-focus position. Then, the next position is generated by dividing the distance into m equal parts with width d/m and setting the new position as $\text{int}(d/m)$ apart from the GT until reaching

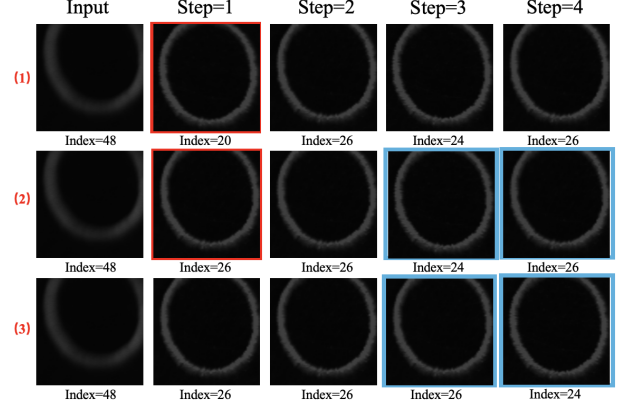


Figure 10. Visualization of Algorithm 1: (1) the original trajectory given; (2) the position of the image in the red rectangle index=20 is symmetrically mirrored from the in-focus position index=23 (GT) to the new position index=26; and (3) resort to the positions.

Algorithm 3 Expert trajectory generation 3.

Input: $\mathbb{S} = \{(s_0, g(\cdot), GT)\}_N$, m and n ;

Output: \mathcal{D}^E ;

```

1:  $\mathcal{D}^E = \emptyset$ ;
2: for all  $(s_0, g(\cdot), GT)_i \in \mathbb{S}$  do
3:    $k_0 \leftarrow g_i^{-1}((s_0)_i)$ ;
4:    $d = k_0 - GT_i$ ;
5:   for all  $j = 1, 2, \dots, n-1$  do
6:      $d \leftarrow \text{int}(d/m)$ ;
7:      $k_j \leftarrow GT_i + d$ ;
8:      $s_j \leftarrow g_i(k_j)$ ;
9:      $a_{j-1} \leftarrow k_{j-1} - k_j$ ;
10:  end for
11:   $s_n \leftarrow g_i(GT_i)$ ;
12:   $a_{n-1} = k_{n-1} - GT_i, a_n \leftarrow 0$ ;
13:   $\mathcal{D}^E \leftarrow \mathcal{D}^E \cup \{g(\cdot), GT, (s_0, a_0), \dots, (s_n, a_n)\}_i$ ;
14: end for
15: return  $\mathcal{D}^E$ .

```

the length of the original trajectory.

In comparison, Algorithm 1 is used to fix the trajectories in which overshooting occurs at the first step or the lens hunting around the in-focus position, since the images corresponding to the lens around the in-focus position have little difference in sharpness and stopping at these positions is more acceptable than hunting around the in-focus position. Algorithm 2 generates trajectories that are difficult to focus on (e.g., no texture) since exploration cannot give more useful information in these scenes. Algorithm 3 can be used to generate any expert trajectories, which is mainly adopted in this paper to generate expert trajectories where the images are sharp.

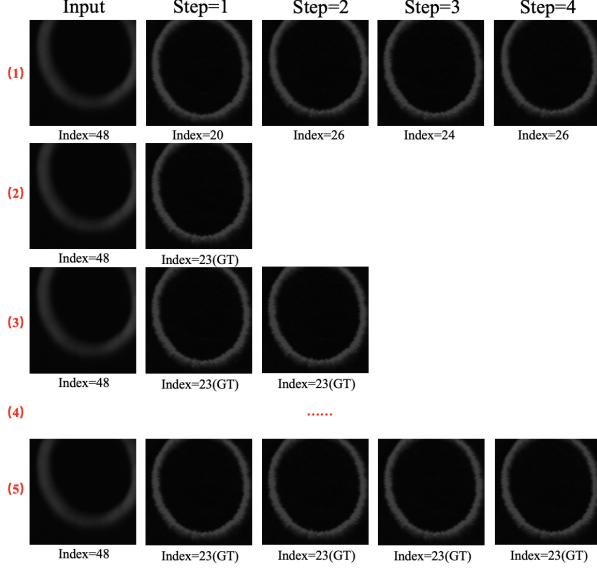


Figure 11. Visualization of Algorithm 2: (1) the origin trajectory given; (2) keep the initial position unchanged; (3) set the in-focus position to the next position; (4) repeat (3) until max steps; (5) get the new trajectory.

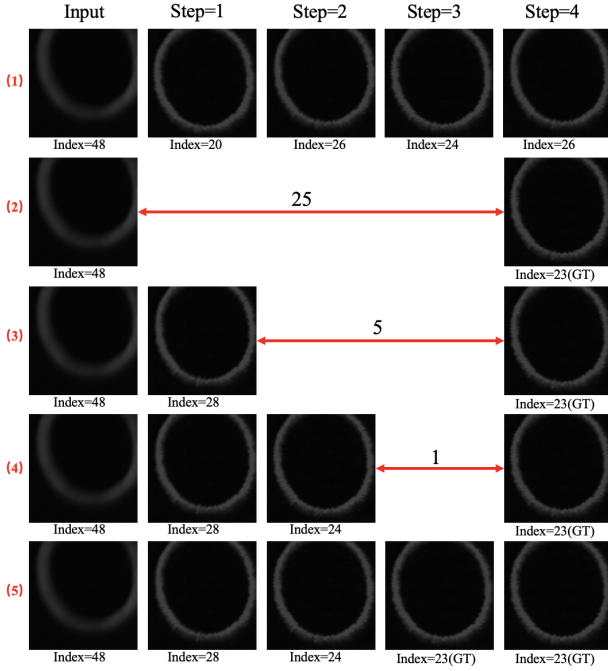


Figure 12. Visualization of Algorithm 3: (1) the origin trajectory given; (2) calculate the distance between the initial and in-focus positions, which is 25; (3) divide the distance=25 into five equal parts with a width=5, and set the new position to be the “width=5” apart from the GT; (4) repeat (3) until max steps; (5) get the new trajectory.

Table 6. Effects of expert regularization.

regularization coefficient	$= 0 \uparrow$	$\leq 1 \uparrow$	$\leq 2 \uparrow$	$\leq 4 \uparrow$	MAE↓	RMSE↓
$\lambda = 1$	0.029	0.050	0.082	0.153	17.213	20.826
$\lambda = 1 \times 10^{-3}$	0.336	0.711	0.839	0.915	2.146	5.827
$\lambda = 1 \times 10^{-7}$	0.023	0.067	0.111	0.191	15.444	19.027

Table 7. Effects of focus hunting penalty.

FH penalty	steps	$= 0 \uparrow$	$\leq 1 \uparrow$	$\leq 2 \uparrow$	$\leq 4 \uparrow$	MAE↓	RMSE↓	FH↓
$c = -1.5$	1	0.336	0.711	0.839	0.915	2.146	5.827	NA
	2	0.439	0.808	0.896	0.941	1.755	5.597	0.170
	3	0.450	0.816	0.900	0.943	1.711	5.532	0.175
	4	0.449	0.818	0.901	0.944	1.694	5.479	0.178
$c = -0.5$	1	0.314	0.688	0.823	0.905	2.448	6.555	NA
	2	0.420	0.798	0.892	0.937	1.868	5.784	0.265
	3	0.427	0.803	0.896	0.940	1.825	5.747	0.307
	4	0.428	0.805	0.897	0.940	1.824	5.765	0.316
$c = -10$	1	0.251	0.601	0.763	0.863	3.352	8.083	NA
	2	0.349	0.735	0.851	0.907	2.715	7.712	0.156
	3	0.350	0.739	0.854	0.907	2.659	7.567	0.157
	4	0.350	0.739	0.854	0.907	2.659	7.567	0.157

9. Qualitative Results and Analysis

9.1. Additional Analysis

MAE and RMSE. Compared with the multi-step predictions obtained by Deep Learning (DL)-based methods, the one-step prediction of the DRL-based method achieves a lower RMSE with higher MAE. As illustrated in Fig. 6, DL-based methods occasionally misidentify the correct in-focus positions, whereas DRL-based methods consistently provide more accurate predictions. Consequently, DL-based methods exhibit higher accuracy within the range of ≤ 4 , but the incorrect predictions deviate significantly from the in-focus positions, resulting in a lower MAE with a higher RMSE. In contrast, despite the DRL-based methods having a lower accuracy within the same range (≤ 4), their incorrect predictions remain closer to the in-focus positions.

9.2. Additional qualitative results

Fig. 13, Fig. 14 and Fig. 15 demonstrate more trajectories of focus hunting, and we provide the corresponding .gif files in our supplementary to visualize the focus hunting more vividly. Fig. 16 illustrates additional failure cases of our models.

9.3. Additional ablation Study

Effects of regularization coefficient. Tab. 6 shows the effects of the regularization coefficient λ in expert trajectory regularization. When the regularization coefficient $\lambda = 1$, the accuracy decreases significantly. This is because when the λ is too large, the effect of the expert trajectory is too strong, and the learning process tends to be imitation learning rather than reinforcement learning. Owing to the distributional shift between the dataset and the actual test environment [32], the system’s performance significantly degrades. When the regularization coefficient $\lambda = 1 \times 10^{-7}$, the accuracy still decreases significantly. This is because

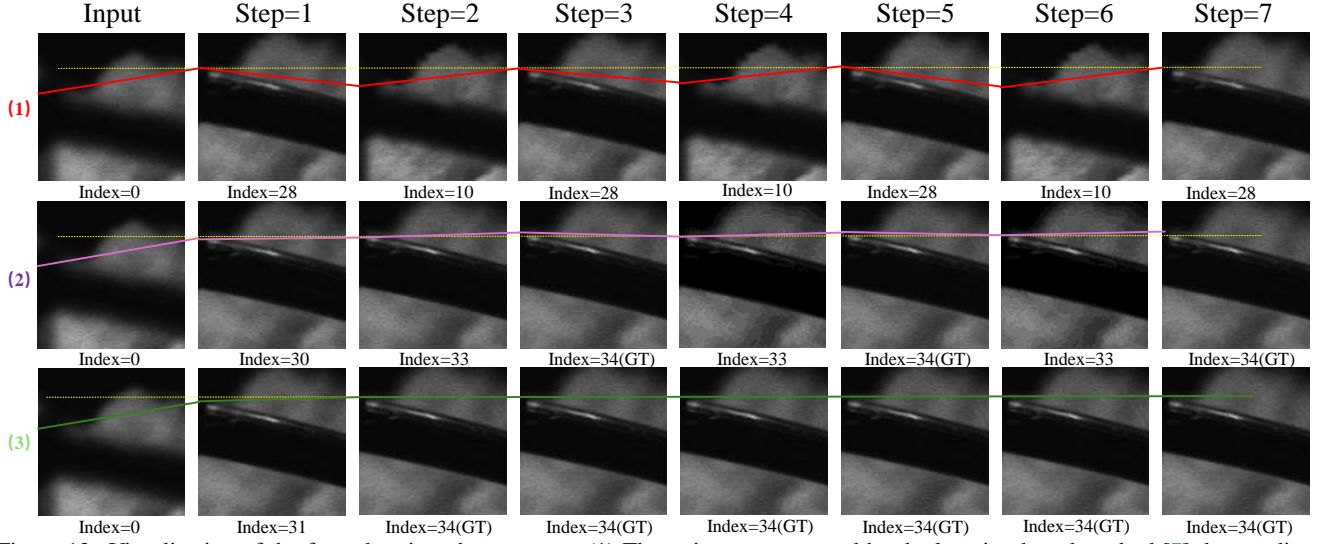


Figure 13. Visualization of the focus hunting phenomenon. (1) The trajectory generated by the learning-based method [7] that predicts the lens position directly. (2) The trajectory generated by the learning-based method [7] that predicts the relative movement of lens. (3) The trajectory generated by our DRL-based method. It can be seen that the FoV and sharpness in trajectory (1) change more significantly, while the lens terminates at the GT after Step 3 in trajectory (2), and the lens terminates at the GT after Step 2 in trajectory (3).

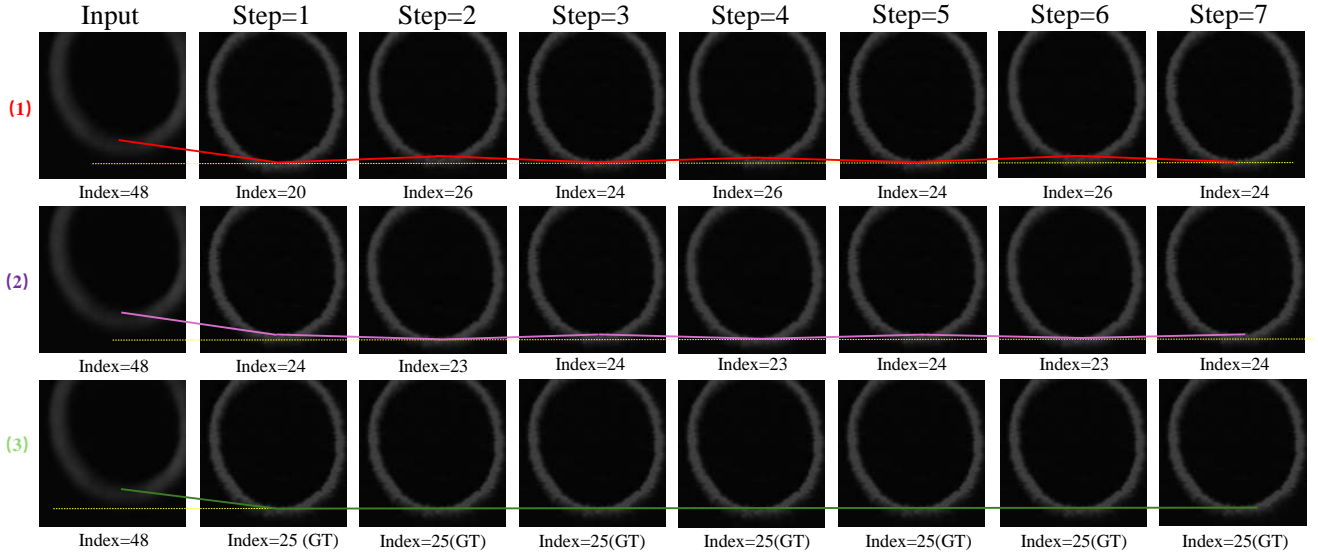


Figure 14. Visualization of the focus hunting phenomenon. (1) The trajectory generated by the learning-based method [7] that predicts the lens position directly. (2) The trajectory generated by the learning-based method [7] that predicts the relative movement of lens. (3) The trajectory generated by our DRL-based method. It can be seen that focus hunting can happen even in scenes with one disparity, leading to the shaking of target objects.

when the λ is too small, the effect of the expert trajectory is too weak, and the learning process tends to be vanilla DRL without expert regularization.

Effects of focus hunting penalty. Tab. 7 demonstrates the effects of focus hunting penalty in Eq. (5). It can be seen that when the focus hunting penalty term is small, the suppressing effects of RL are low, leading to a higher FH rate. When the penalty term is high, the suppressing effects of RL are too strong, leading to no backward movement of the lens and a low FH rate with low accuracy.

10. Limitations and Future Works

While showing state-of-the-art performance, our work may be further improved in the future by addressing the following limitations: 1) The adopted dataset is static, so the performance in dynamic scenes remains unknown. 2) The design of the reward function is simple, while the human preference in AF tasks is hard to model. More advanced technologies, such as reinforcement learning from human feedback (RLHF), may be further leveraged to help design a better reward function and refine the trained model.

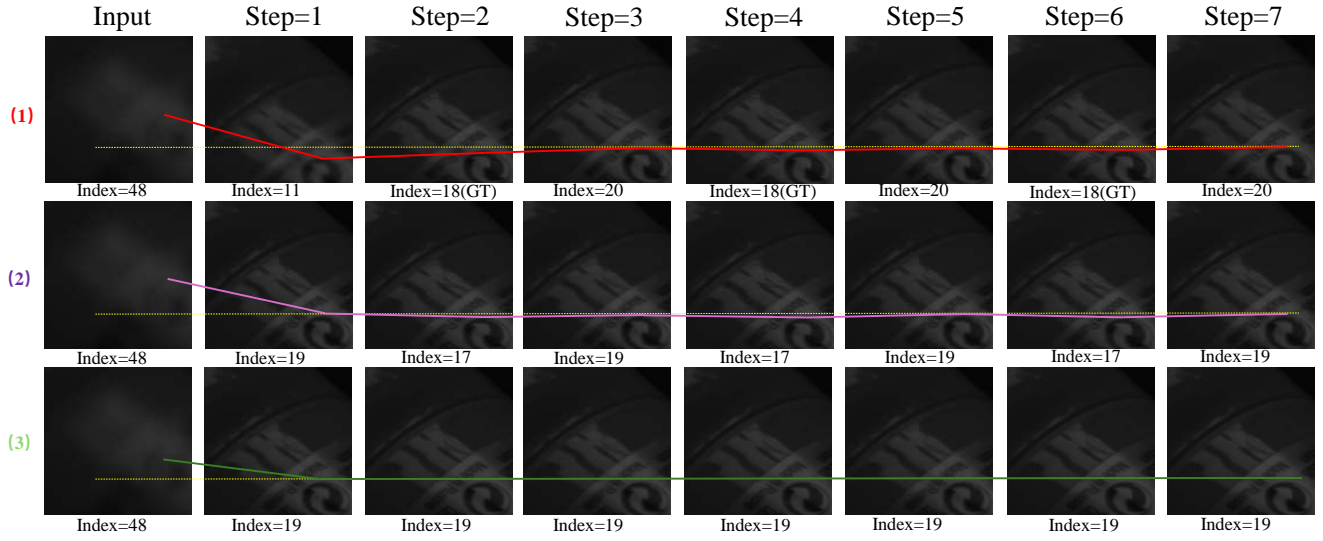


Figure 15. Visualization of the focus hunting phenomenon. (1) The trajectory generated by the learning-based method [7] that predicts the lens position directly. (2) The trajectory generated by the learning-based method [7] that predicts the relative movement of lens. (3) The trajectory generated by our DRL-based method. Although the learning-based method [7] can predict the in-focus position, focus hunting still occurs.

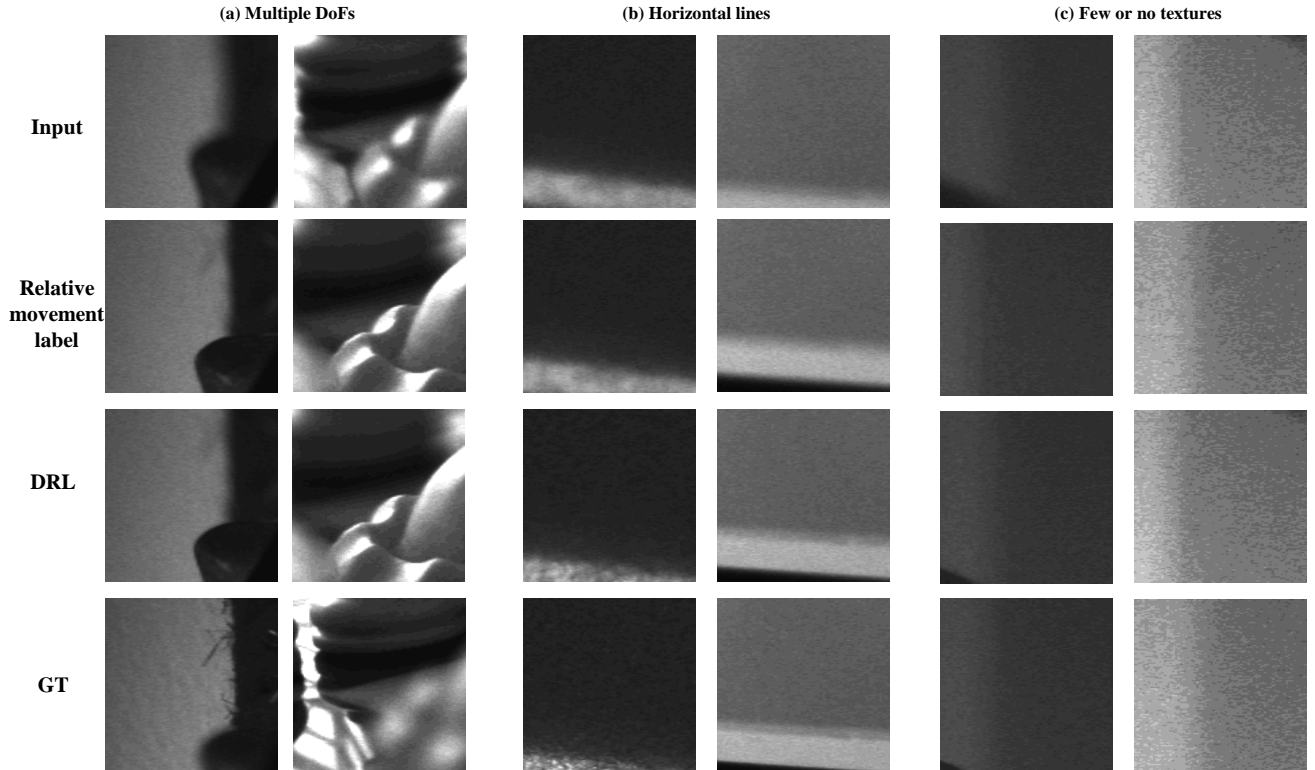


Figure 16. Additional failure cases of our models, where only the *left* image of DP data is visualized.

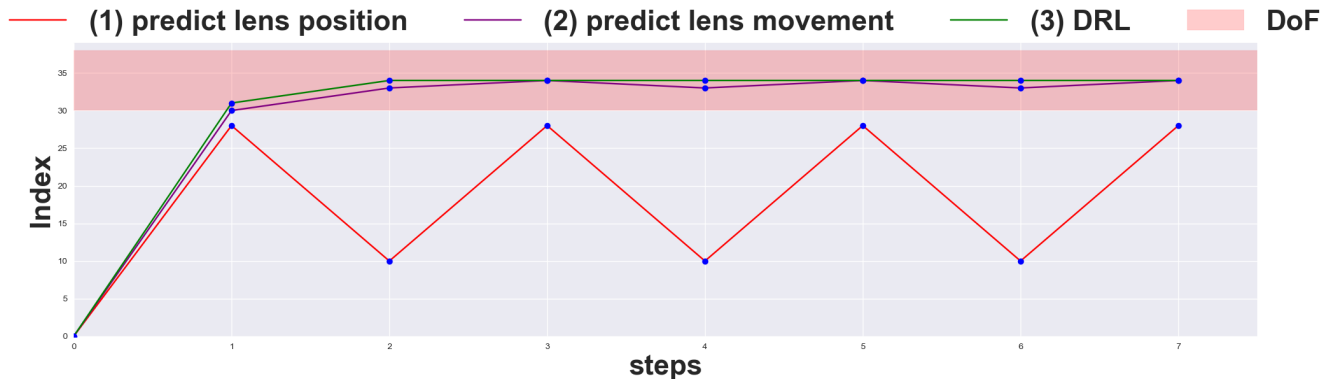


Figure 17. Trajectories of Fig. 13 generated by predicting the lens position [7] (red line), predicting the relative movement (purple line), and DRL (green line). The trajectory generated by our DRL is the smoothest, indicating a stable and fast focusing procedure.

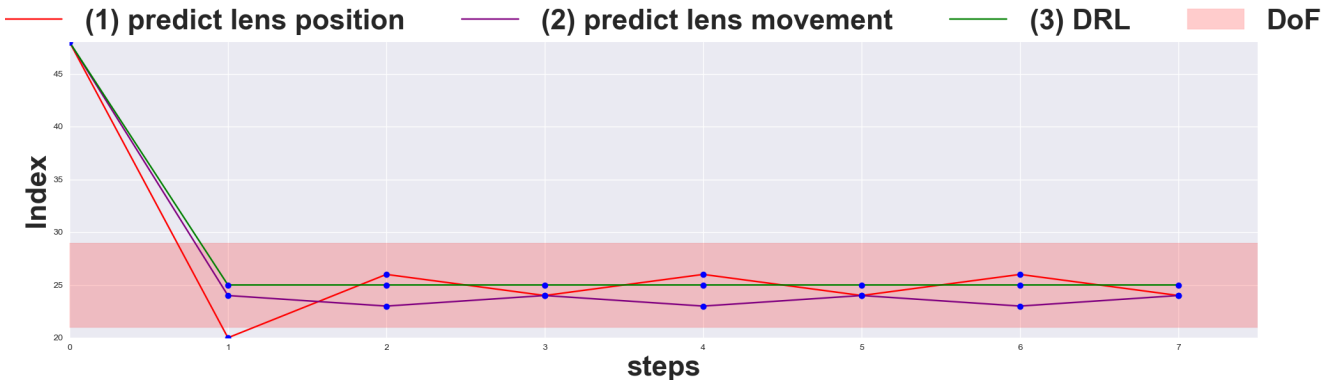


Figure 18. Trajectories of Fig. 14 generated by predicting the lens position [7] (red line), predicting the relative movement (purple line), and DRL (green line). The trajectory generated by our DRL is the smoothest, indicating a stable and fast focusing procedure.

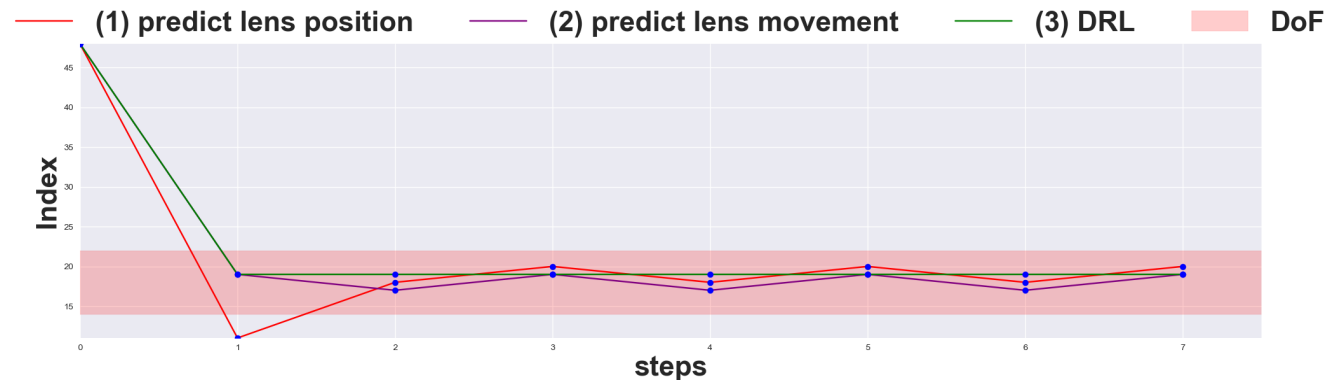


Figure 19. Trajectories of Fig. 15 generated by predicting the lens position [7] (red line), predicting the relative movement (purple line), and DRL (green line). The trajectory generated by our DRL is the smoothest, indicating a stable and fast focusing procedure.