Data-Driven Active Snakes with Reinforcement Learning for Image Segmentation

Rishi Upadhyay Eshanika Ray Nikhil Isukapalli Jakob Reinwald

June 20, 2025

Abstract—Automated image segmentation is crucial in various applications, including medical imaging. Traditional active contour models, or "snakes," are powerful tools for image segmentation that evolve contours towards object boundaries by minimizing an energy function. However, classical snakes often suffer from limitations such as local minima traps, initialization sensitivity, and tedious manual parameter tuning. While deep learning has significantly advanced computer vision, it typically requires large, manually annotated datasets, which are time-intensive and intricate to create, especially in medical contexts. This project proposes a novel approach to address these limitations by developing an active snake controlled by a neural network trained using Reinforcement Learning (RL). This data-driven approach aims to move beyond fixed, hand-crafted energy functions and enable the snake to learn adaptive, context-aware deformation strategies. We also explore imitation learning, training our RL snake to emulate a traditional snake to accelerate basic movement learning. This framework has the potential to provide more robust and intuitive segmentation tools that can leverage global image context in addition to local features. We intend to apply these techniques to instance segmentation tasks on traditional images, with potential extensions to medical images.

1 Introduction

Automated image segmentation holds significant clinical utility in various domains, particularly in medical imaging. However, progress in this area often faces challenges, including the demanding need for high-quality ground-truth annotations. Manual annotation, while possible, is elaborate and time-intensive, thereby limiting large-scale studies. Deep learning methods have generated considerable recent success in computer vision [1], but require a significant amount of labeled samples, which presents a hurdle.

Active contour models, often referred to as "snakes," represent a classic yet powerful approach to image segmentation, introduced by Kass et al. [2]. These deformable splines evolve toward object boundaries by minimizing an energy function, balancing internal forces (for smoothness) and external forces (for attraction to image features like edges). While they provide smooth, interpretable boundaries, classical snakes suffer from several limitations: they are prone to getting trapped in local minima, are sensitive to initialization, and require extensive manual parameter tuning [3, 4].

To address the annotation bottleneck in deep learning and the rigidity of traditional active contours, this project introduces a novel learning framework. We propose a data-driven active snake model, controlled by a neural network trained via Reinforcement Learning (RL) [5]. Unlike classical snakes with fixed deformation rules, our RL-controlled snake learns to adapt its strategy based on image context and reward feedback. The RL agent outputs dynamic offsets for each control point of the snake, and is guided by reward signals derived from reductions in classical energy terms.

Additionally, we integrate imitation learning [6] into our framework, enabling the agent to first mimic the behavior of a traditional snake before transitioning to reward-based policy learning. This allows the system to learn stable, basic deformation behaviors early in training and accelerates convergence.

This formulation opens the door to segmentation tools that are more intuitive, adaptable, and capable of leveraging both global context and local structure without requiring extensive supervision or handcrafted design.

Contributions- Our key contributions are:

- We formulate active contour evolution as an RL control problem, using classical snake energy as a reward signal.
- We incorporate imitation learning to bootstrap the agent's behavior from a traditional snake expert.
- We validate the framework on synthetic and natural images, demonstrating the potential of learned snake control.

2 RELATED WORK

The evolution of image segmentation methods, particularly those involving deformable models, has seen a significant transition from classical variational approaches to data-driven and deep learning paradigms. Our work is situated at the intersection of these two threads, aiming to retain the interpretability of classical models while enhancing adaptability through reinforcement learning.

2.1 Classical Active Contours

The foundational work by Kass et al. [2] introduced active contours, or "snakes," as energy-minimizing deformable splines that evolve toward object boundaries. The classical snake model balances two types of forces:

Internal Forces: These maintain smoothness and regularity of the contour through:

- *Continuity term*: Penalizes stretching, acting like a rubber band pulling points together
- *Curvature term*: Penalizes bending, encouraging smooth curves

External Forces: These attract the contour to image features, primarily image gradients and edges. The snake evolves by minimizing the total energy functional defined over the continuous curve $v(s)$:

$$\begin{aligned} E_{\text{snake}} = \int_0^1 & (E_{\text{internal}}(v(s)) \\ & + E_{\text{image}}(v(s)) + E_{\text{constraint}}(v(s)))\, ds \end{aligned} \tag{1}$$

where E_{internal} penalizes high curvature, E_{image} draws the curve toward image gradients, and $E_{\text{constraint}}$ optionally enforces task-specific forces.

Key Limitations: Despite their foundational impact and intuitive formulation, classical snakes suffer from several critical limitations:

- *Local minima traps*: Energy minimization often gets stuck in suboptimal solutions
- *Initialization sensitivity*: Performance heavily depends on proximity to target boundaries
- *Manual parameter tuning*: Requires extensive hand-tuning of energy weights for different applications
- *Limited capture range*: External forces have limited influence beyond immediate neighborhood

These models were extended to level sets and geometric deformable models in 2D and 3D medical imaging, often requiring strong priors or manual initialization [2]. Their deterministic nature limits flexibility in adapting to noisy or complex image conditions, motivating the development of more sophisticated learning-based approaches.

2.2 Modern Active Contours

The field has evolved significantly to address the limitations of classical snakes, with a clear progression from hand-crafted energy functions to learned features. Two key developments exemplify this transition:

Deep Active Contours (2016): This work [7] introduced the use of convolutional neural networks (CNNs) to predict displacement vector fields that guide contour evolution. Rather than relying on hand-crafted image gradients, the CNN learns to extract relevant features and predict optimal deformation directions. This approach removes the need for manual energy design while retaining the interpretable deformable boundary representation of classical snakes.

Deep ContourFlow (2024): More recently, Deep ContourFlow [8] proposed an unsupervised contour evolution framework that learns to align contours without explicit supervision. This approach is particularly valuable for medical applications where labeled data is scarce and expensive to obtain. The method focuses on learning contour dynamics from the data itself, reducing dependence on manual annotations.

Overall Progress: The overarching trend has been moving from hand-crafted energy functions → learned features → unsupervised learning. This evolution addresses many traditional limitations:

- *Feature learning*: CNNs can capture complex image patterns beyond simple gradients
- *Reduced manual tuning*: Learning-based approaches adapt parameters automatically
- *Domain adaptation*: Learned features can generalize across different image types

Despite these advances, current approaches still rely on fixed policies or architectures that do not adapt based on context or history. Most methods lack explicit mechanisms for exploration, temporal credit assignment, or context-dependent strategy selection. This gap in adaptive decision-making is what our reinforcement learning approach aims to fill, enabling snakes to learn flexible, context-sensitive deformation strategies rather than following predetermined rules.

2.3 DeepSnake - Our Baseline Method

DeepSnake [9] represents the current state-of-the-art in learning-based active contours and serves as our primary baseline. This method combines deep learning with classical snake formulations through several key innovations:

Architecture: DeepSnake models the contour as a cycle graph and employs circular convolutions specifically designed for the cyclic structure of closed contours. The network iteratively deforms an initial contour to match object boundaries, implementing classic snake algorithms within a learning-based framework.

Pipeline: The method operates in two stages: (1) an object detector provides initial polygon predictions, and (2) iterative contour refinement using the circular convolution network. This allows end-to-end learning of boundary evolution while maintaining the interpretable structure of traditional snakes.

Performance: DeepSnake achieves impressive real-time performance of 32.3 fps on 512×512 images while maintaining competitive accuracy on instance segmentation benchmarks. This demonstrates the practical viability of learning-based snake approaches.

Limitations: Despite its success, DeepSnake remains constrained by a fixed learned deformation policy that cannot adapt based on contour history, region complexity, or failure scenarios. The model lacks explicit mechanisms for exploration, temporal credit assignment, or context-dependent strategy selection—capabilities that are fundamental to reinforcement learning approaches. Additionally, the deterministic nature of its deformation strategy limits its ability to recover from poor intermediate states or adapt to novel image conditions not seen during training.

2.4 Reinforcement Learning in Image Segmentation

While reinforcement learning has been extensively applied to various computer vision tasks, its application to image segmentation remains relatively underexplored, particularly for direct contour control.

Early RL Applications - Medical Segmentation: Sahba et al. [10] pioneered the use of RL in segmentation by proposing a Q-learning framework to optimize post-processing parameters for medical segmentation. Their approach used Q-learning to find optimal thresholds and parameter configurations, demonstrating the key insight that segmentation could be formulated as a control problem where an agent learns to make optimal decisions based on the image context.

Recent RL Segmentation Methods: More recent work has expanded RL applications in segmentation:

- *Multi-step Medical Segmentation*: Tian et al. [11] formulated segmentation as a Markov Decision Process (MDP) using Deep Deterministic Policy Gradient (DDPG) algorithms, enabling sequential decision-making for complex anatomical structures
- *Reinforced Active Learning*: Casanova et al. [12] applied RL to reduce annotation requirements, with their Deep Q-Network (DQN) approach achieving 30% reduction in labeled data needed while maintaining segmentation quality on Cityscapes dataset
- *Medical Image Analysis*: Hu et al. [13] developed comprehensive RL frameworks for medical image analysis, demonstrating applications across detection, classification, and segmentation tasks
- *Parameter Optimization*: Most existing methods focus on tuning hyperparameters, selecting thresholds, or optimizing post-processing steps rather than directly controlling segmentation boundaries

Despite these advances, most existing RL approaches operate at a high level, optimizing parameters or selecting regions, rather than directly controlling contour deformation. To the best of our knowledge, no prior work has formulated active contour evolution as an RL control problem where an agent directly learns to move individual control points.

Our approach departs from heuristic energy minimization by learning a policy over control point displacements, driven by classical energy terms as rewards. This allows a data-driven agent to learn flexible, context-sensitive deformation strategies, bridging the gap between traditional snake energy formulations and modern RL capabilities.

3 METHODOLOGY

Our work aims to develop an active snake controlled by a neural network trained with Reinforcement Learning (RL). This section outlines our proposed methodology, detailing the design of our RL environment, neural network architecture, and distinct learning approaches.

3.1 RL Environment Design

The foundation of our approach involves defining an RL environment where a neural network controller outputs offsets at each timestep to deform snake points. The network receives inputs ranging from local image information at snake points to features from the entire image for global context.

3.1.1 Observation Space

Determining the dimension and structure of the observation space is crucial for effective learning, ensuring sufficient information is available while minimizing unnecessary dimensions. Initially, observations included only snake point locations and local pixel values. However, this proved insufficient due to a lack of context, leading us to enrich the state by including image data from the snake's current bounding box. This provides a more comprehensive scope of relevant image information. Our observations now consist of two main components: an image input tensor, typically a Region of Interest (ROI) from the input image, and a vector input tensor. The image input is represented as a 4D tensor with shape (N, C, H, W) (batch, channels, height, width). The vector input, combining features like Center of Mass (COM) or traditional snake point data, is a 2D tensor with shape (N, D_v), where D_v is the vector's feature dimension.

3.1.2 Action Space

Our initial design for the action space involved the model outputting 2D offsets for each vertex of the snake, aiming for fine-grained control. We realized this created a high-dimensional space, challenging for quick learning. We are now outputting a 1D offset for each point, applied along the snake's normal vector. This offers a more constrained action space for the RL agent to master while allowing expressive deformations.

3.1.3 Episode Termination

Our RL environment requires a clear definition for episode termination, which is when a complete sequence of events, from initial state to end state, has occurred. Our solution involves two main conditions: if the snake's internal energy exceeds a certain threshold (indicating excessive deformation like overstretching or self-intersecting), or if its total energy has not changed significantly over several iterations (indicating stuckness or convergence). Fulfillment of either condition concludes the episode.

3.1.4 Reward Design

Reward design is a key aspect of our RL setup, guiding the agent's learning process. Initial attempts used the negative sum of traditional internal and external energies, aiming to reward actions that lower total energy. However, issues arose due to the reward signal being consistently negative and its magnitude varying greatly, leading to inconsistent learning. An alternative considered was rewarding the improvement in energy from one timestep to the next, directly encouraging the snake to take actions that lower its overall energy. However, this risked trapping the snake in local minima if small moves didn't decrease energy. Relying on energy-derived signals proved to be challenging, leading us to consider other learning techniques.

In our current implementation, the reward is dynamically calculated based on the availability of ground truth annotations. For cases where ground truth is available, the reward combines an exponential term reflecting proximity to the ground truth with the difference between external and internal energies (i.e., $\exp(-0.1 \times \text{mean\_abs\_diff}) + E_{\text{external}} - E_{\text{internal}}$). For scenarios without ground truth (e.g., toy problems where only general convergence is desired), the reward is based on the improvement of a scaled energy sum over the previous timestep (i.e., $(0.1 \times (E_{\text{external}} + \exp(-0.1 \times E_{\text{internal}}))) - \text{last\_reward}$). A critical component of our reward structure is the penalty for instability: if the snake's internal energy exceeds a predefined threshold, a significant negative reward (e.g., -5.0) is assigned, and the episode terminates. Balancing these reward components remains an ongoing challenge in guiding optimal snake behavior.

Neural Network Architecture

For our deep learning components, we utilize distinct architectures based on the `model_type` configuration, namely Multi-Layer Perceptron (MLP) and Convolutional Neural Network (CNN).

Multi-Layer Perceptron (MLP)

The `ActorCritic` network serves as our MLP architecture when observations are primarily vector-based or when image inputs are flattened for direct consumption. This network comprises a sequential stack of fully connected (`nn.Linear`) layers with ReLU activations (`nn.ReLU()`), producing an output action via a Tanh activation (`nn.Tanh()`). The general structure is:

nn.Linear(*state_dim*, 64) → nn.ReLU() → nn.Linear(64, 64) → nn.ReLU() → nn.Linear(64, *action_dim*) → nn.Tanh()

The `state_dim` is dynamically calculated to match the total feature count of the flattened or concatenated vector input.

Convolutional Neural Network (CNN)

For robust image processing, particularly with Region of Interest (ROI) observations, we employ a CNN-based `CNNActorCritic` architecture. This model incorporates a `cnn_base` for image feature extraction, typically comprising sequential `nn.Conv2d` layers and `nn.MaxPool2d` layers, culminating in an `nn.Flatten()` operation. A separate MLP, the `vector_feature_processor`, handles non-image vector inputs. Features from both the `cnn_base` and `vector_feature_processor` are then concatenated and processed by `shared_layers` before branching into distinct `actor_head` and `critic_head` components. This design allows the network to process raw image data (e.g., 200x200 pixels with C channels, shape $(N, C, 200, 200)$) alongside other numerical features effectively.

Direct Reinforcement Learning (PPO)

Our primary approach for learning the snake's control policy is Direct Reinforcement Learning, specifically using the Proximal Policy Optimization (PPO) algorithm, implemented in `PPOTrainer`. PPO directly optimizes the agent's policy by interacting with the environment and maximizing cumulative reward through exploration, rather than relying solely on expert demonstrations. It employs both an actor network (policy) and a critic network (value function) to guide the learning process. This allows the agent to potentially discover novel strategies and avoid replicating expert imperfections, such as those arising from limited expert exploration.

Imitation Learning

Our project also incorporates Imitation Learning, specifically Behavioral Cloning (BC), managed by `BCTrainer`. In this approach, the model is trained to replicate an "expert". We achieve this by having our RL snake emulate a traditional snake: at each optimization step, the action of the traditional snake (the "expert action") serves as a supervised learning target for the neural network. For the purposes of this project, our expert snake used the traditional internal energy and the edge-finding external energy. The core of the training involves minimizing the Mean Squared Error (MSE) between the predicted action ($\text{action}_{\text{predicted}}$) and the expert action ($\text{action}_{\text{expert}}$), which is computed as:

$$L_{\text{BC}} = \frac{1}{N} \sum_{i=1}^{N} ||\text{action}_{\text{predicted},i} - \text{action}_{\text{expert},i}||^2$$

where N is the number of samples in the batch. This aims to help the snake learn basic movements more quickly, after which broader fine-tuning can occur.

4 EXPERIMENTS/RESULTS

4.1 Classical Snake Benchmark

For the classical snake, the `main.py` script iterates through various predefined mask shapes (e.g., circle, rectangle, triangle). For each shape, it sets up an initial snake

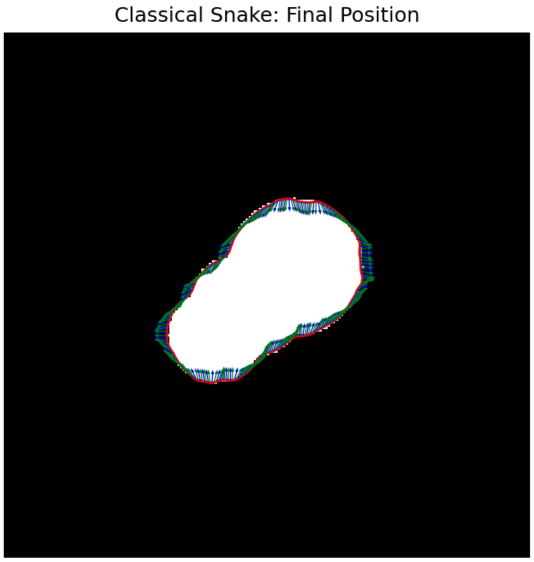


Figure 1: Results from running a classical snake on a multi-circle model. The snake is able to perfectly converge to the proper image segmentation.

contour that is strategically placed to intersect the target object's boundaries, ensuring non-zero external forces. The classical snake model is then optimized for a fixed number of iterations. The initial image, final segmentation, and gradient fields are then saved. An example of one of these runs can be seen in 1.

4.2 Classical Snake Performance

The classical snake model demonstrated perfect convergence for simple, smooth shapes like circles and multi-circles when ‘num_snake_points‘ was set to a high enough value. This behavior was achieved with parameters **alpha=1.0**, **beta=0.001**, and **gamma=1.0**. The low ‘beta‘ value allowed sufficient flexibility while maintaining coherence. However, for more complex shapes such as triangles and stars, the classical snake took more time to converge to the precise boundaries, sometimes collapsing or maintaining a rounded shape due to its inherent preference for smoothness.

4.3 RL Snake Training and Evaluation

These experiments are systematically controlled by a comprehensive `rl_snake_params` dictionary defined within `main.py`. This dictionary serves as a central configuration hub, defining various aspects of the RL training and evaluation process. It specifies key parameters for snake mechanics (e.g., `num_snake_points`, `alpha`, `beta`, `gamma`), training hyperparameters (e.g., `num_settings` for total episodes, `update_freq`, `save_freq`), and crucial model architecture choices such as `model_types` (“mlp”, “cnn”) and `obs_types` (e.g., “com,roi”, “trad”). Most importantly, it allows us to programmatically generate results for each of our implemented `trainer_algorithms` (“ppo”, “bc”), and the set of target `shapes` to be included in the evaluation, which are obtained via `get_image_setting_functions`, ensuring consistency and reproducibility across all experimental runs. Training data is logged to Weights & Biases (WandB). After training, a specific checkpoint of the trained model is loaded and evaluated across multiple episodes on various image settings.

In total, we produced end-to-end model training and evaluation scenarios for a total of 12 different models. For each trainer algorithm(direct-RL, imitation), we trained the model on 1 of the 6 images, and evaluated each training on all 6 of the images, including the one it was trained on.

4.4 RL Snake Performance

Initial training runs for the RL snake models struggled significantly, with the snake often collapsing to the center or not moving at all. This was traced to several subtle but critical bugs in the observation and external force calculation pipelines, including `GradientExternalEnergy` failing to compute non-zero forces due to snake point misplacement or zero-gradient regions, mismatch in coordinate systems where the ground truth action function (`gt_action_fn`) received normalized coordinates but `bilinear_interpolate` expected pixel coordinates, and `get_com_obs` not normalizing its output leading to wildly scaled features.

After extensive debugging, these fundamental issues were resolved, allowing the snake to finally “feel” the gradients and move across the image. Subsequent runs showed the snake's boundary starting to deform from its initial circle, although it often remained largely circular even when targeting complex shapes like a star. The `bc_loss` was initially very high but showed rapid initial drop-off, indicating some learning, though it would often plateau at relatively high values. Rewards were occasionally negative due to the internal energy exceeding a set threshold, leading to early episode termination.

In total, we performed end-to-end model training and evaluation scenarios for various configurations. We specifically trained MLP models across all six primary image shapes (circle, rectangle, triangle, star, ellipse, and multi-circle) using both the Proximal Policy Optimization (PPO) and Behavioral Cloning (BC) algorithms. Subsequently, every single trained model was then evaluated against all 6 images to assess basic convergence and performance stability and zero-shot applicability.

Overall, across these evaluations, the Behavioral Cloning (BC) models performed better on average compared to PPO models. Preliminary PPO results (such as for circles) showed promise after initial issues were resolved, and we were able to improve the PPO results, suggesting that direct RL has the potential to learn effective policies with further optimization. However, PPO models generally performed poorly across diverse image types, likely due to the high-dimensional action space and the challenge of sparse or noisy reward signals. The imitation learning models, while not always perfect, were able to learn much faster and often largely maintained their shape, showing an affinity for edges, as can be seen in qualitative examples.

One of our ideal runs demonstrating strong performance for an MLP BC model on a circle is illustrated in Figure 2, with its corresponding BC loss and reward progression shown in Figures 3 and 4. While many other runs exhibited negative rewards or problematic loss curves, these figures exemplify the best outcomes achieved, demonstrating the potential for convergence and effective segmentation under ideal conditions.

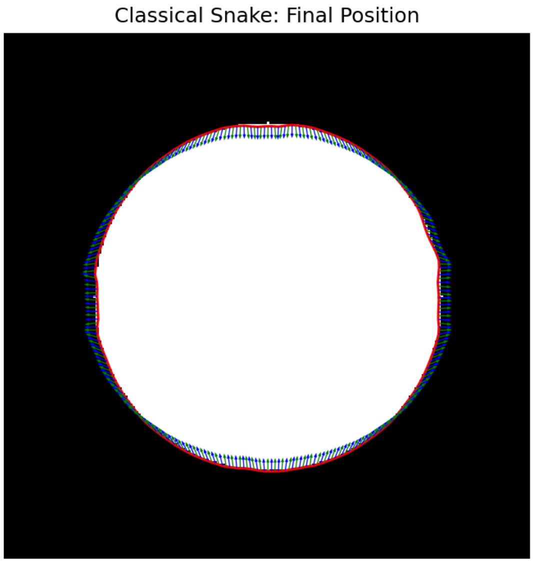


Figure 2: Final snake segmentation for MLP BC model on a circle.

4.5 Black-and-white Shapes

Due to the difficulties of training the RL models well, we first tested our snakes on simpler black-and-white images which were synthetically generated. Examples of these images can be seen in Fig. 7. As mentioned in the imitation learning section, we trained our models using a snake designed to find edges in the image. For the experiments shown in Fig. 7 and Fig. 5, we trained our models on the multi-circle images. For these images, we randomly place up to 5 white circles on a black image. These combine to generate a psuedo-random shape. Results for both the PPO and Imitation learning based models were obtained, showing much better performance by the imitation learning. The PPO model performs poorly across all types of images, likely due to the high-dimensional action space and the sparse rewards. With longer term training, the PPO models may improve, but it cannot be guaranteed. The imitation learning models are able to learn much faster in the same training times, as can seen from the figure. They largely maintain their shape and show an affinity for edges.

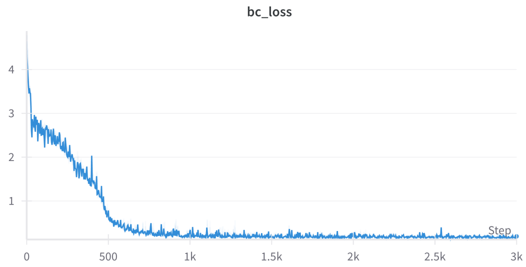


Figure 3: Behavioral Cloning Loss curve for MLP BC model on a circle.

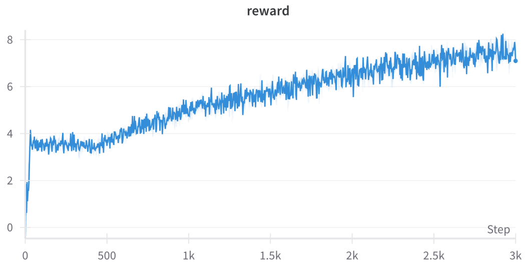


Figure 4: Reward progression for MLP BC model on a circle.

There were some ideal training scenarios found, such as the imitation learning run shown in

4.6 Real-World Images

To evaluate the generality of our trained snakes, we also applied to real-world images. An example of this is shown in Fig. 8. Although the snake exhibited limited performance, it does show some signs of latching onto edges even in a very different type of image.

5 Discussion

The experiments highlight the comparative strengths of classical and learning-based active contour models. Classical snakes performed well on smooth shapes such as circles and multi-circles with tuned parameters ($\alpha = 1.0$, $\beta = 0.001$, $\gamma = 1.0$), but failed on complex shapes like stars and triangles due to oversmoothing and limited capture of high-curvature regions. The PPO-based reinforcement learning agent struggled to learn effective policies. The large action space (300 dimensions), sparse reward signals, and non-convex optimization landscape resulted in unstable convergence, consistent with known challenges in high-dimensional continuous control. In contrast, the behavioral cloning (BC) model, trained on expert snake trajectories, showed stable training and generalization across previously unseen shapes. Dense supervision helped the model learn transferable deformation pat-

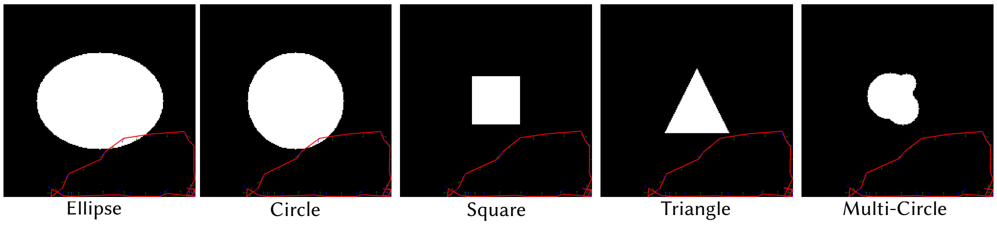


Figure 5: Results from a model trained with PPO and the energy-based reward. The model was trained on only multi-circle images, and was applied zero-shot to the other examples. Due to the high-dimensional action-space and sparse external rewards, we find our snakes perform very poorly across different scenarios, including the training scenario of multi-circles. Improved results can be found in images like 6

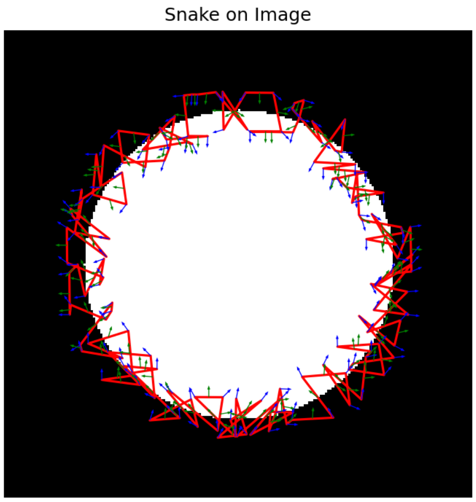


Figure 6: Improved initial PPO results from training on a circle.

terns from synthetic shapes. BC also required fewer updates and exhibited faster convergence than PPO, though both models failed to generalize to natural images.

This work explores a relatively novel direction by treating active contour evolution as a neural control problem. Unlike prior approaches that regress energy parameters or approximate potential functions, we directly learn displacement policies. Future extensions could involve hierarchical or graph-based representations to reduce control dimensionality, use of spatial attention to focus on boundary-relevant regions, and integration of diffusion models or reinforcement learning with auxiliary prediction heads for richer supervision. These approaches may enhance generalization to complex, real-world scenarios such as medical image segmentation or scientific imaging tasks.

6 Conclusion

This work presents a unified framework for controlling active contours using reinforcement and imitation learning. By framing snake deformation as a sequential control task, we enable neural agents to learn geometry-aware displacement strategies.

Our contributions include: (1) an RL-compatible environment with structured observation, action, and reward spaces; (2) a comparative analysis showing PPO's limitations under high-dimensional, sparse-reward settings; (3) evidence that behavioral cloning enables stable learning and shape generalization; and (4) architectural comparisons of MLP and CNN-based agents.

While behavioral cloning showed strong performance on synthetic shapes, generalization to natural images remains limited. PPO was hindered by instability and reward sparsity. These findings suggest potential in combining expert supervision with learning-based adaptability.

Future work should explore hierarchical actions, curriculum training, and hybrid reward signals. This framework offers a foundation for interpretable, adaptive segmentation tools in domains such as medical imaging.

References

[1] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[2] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988.

[3] Donna J Williams and Mubarak Shah. A fast algorithm for active contours and curvature estimation. *CVGIP: Image Understanding*, 55(1):14–26, 1992.

[4] Amir A Amini, Terry E Weymouth, and Ramesh C Jain. Using dynamic programming for solving variational problems in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(9):855–867, 1990.

[5] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT Press, Cambridge, 1998.

[6] Takayuki Osa, Joni Pajarinen, Gerhard Neumann, J Andrew Bagnell, Pieter Abbeel, and Jan Peters.

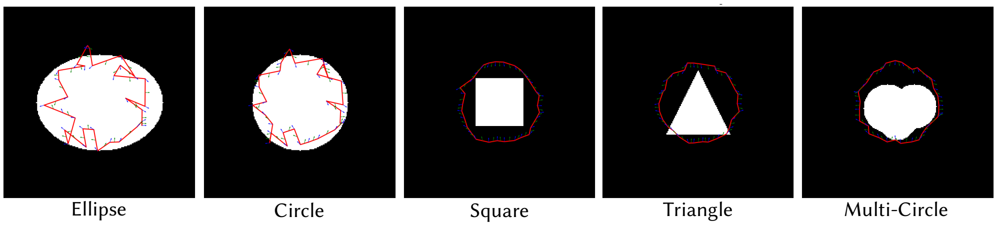
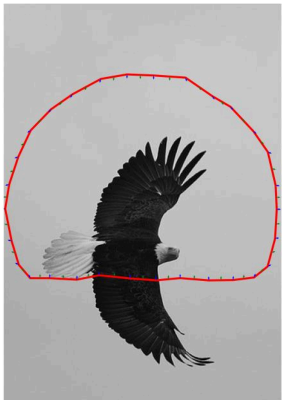


Figure 7: Results from a model trained with imitation learning. The model was trained on only multi-circle images, and was applied zero-shot to the other examples.



Eagle Natural Image

Figure 8: Results from a model trained with imitation learning on a real natural image. The snake was unable to find the overall boundary of the eagle, but has latched on to the bottom of tail and with more iterations, may be able to capture the full boundary.

An algorithmic perspective on imitation learning. *Foundations and Trends in Robotics*, 7(1–2):1–179, 2018.

[7] Christian Rupprecht, Elizabeth Huaroc, Maximilian Baust, and Nassir Navab. Deep active contours. *arXiv preprint arXiv:1607.05074*, 2016.

[8] Antoine Habis, Vannary Meas-Yedid, Elsa Angelini, and Jean-Christophe Olivo-Marin. Deep contourflow: Advancing active contours with deep learning. *arXiv preprint arXiv:2407.10696*, 2024.

[9] Sida Peng, Wen Jiang, Huaijin Pi, Xiuli Li, Hujun Bao, and Xiaowei Zhou. Deep snake for real-time instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8533–8542, 2020.

[10] Farhang Sahba, Hamid R Tizhoosh, and Magdy MA Salama. A reinforcement learning framework for medical image segmentation. In *The 2006 IEEE International Joint Conference on Neural Network Proceedings*, pages 511–517. IEEE, 2006.

[11] Zhiqiang Tian, Xiangyu Si, Yaoyue Zheng, Zhang Chen, and Xiaojian Li. Multi-step medical image segmentation based on reinforcement learning. *Journal of Ambient Intelligence and Humanized Computing*, pages 1–12, 2022.

[12] Arantxa Casanova, Pedro O Pinheiro, Negar Rostamzadeh, and Christopher J Pal. Reinforced active learning for image segmentation. *arXiv preprint arXiv:2002.06583*, 2020.

[13] Mingzhe Hu, Jiahan Zhang, Luke Matkovic, Tian Liu, and Xiaofeng Yang. Reinforcement learning in medical image analysis: Concepts, applications, challenges, and future directions. *Journal of Applied Clinical Medical Physics*, 24(2):e13898, 2023.