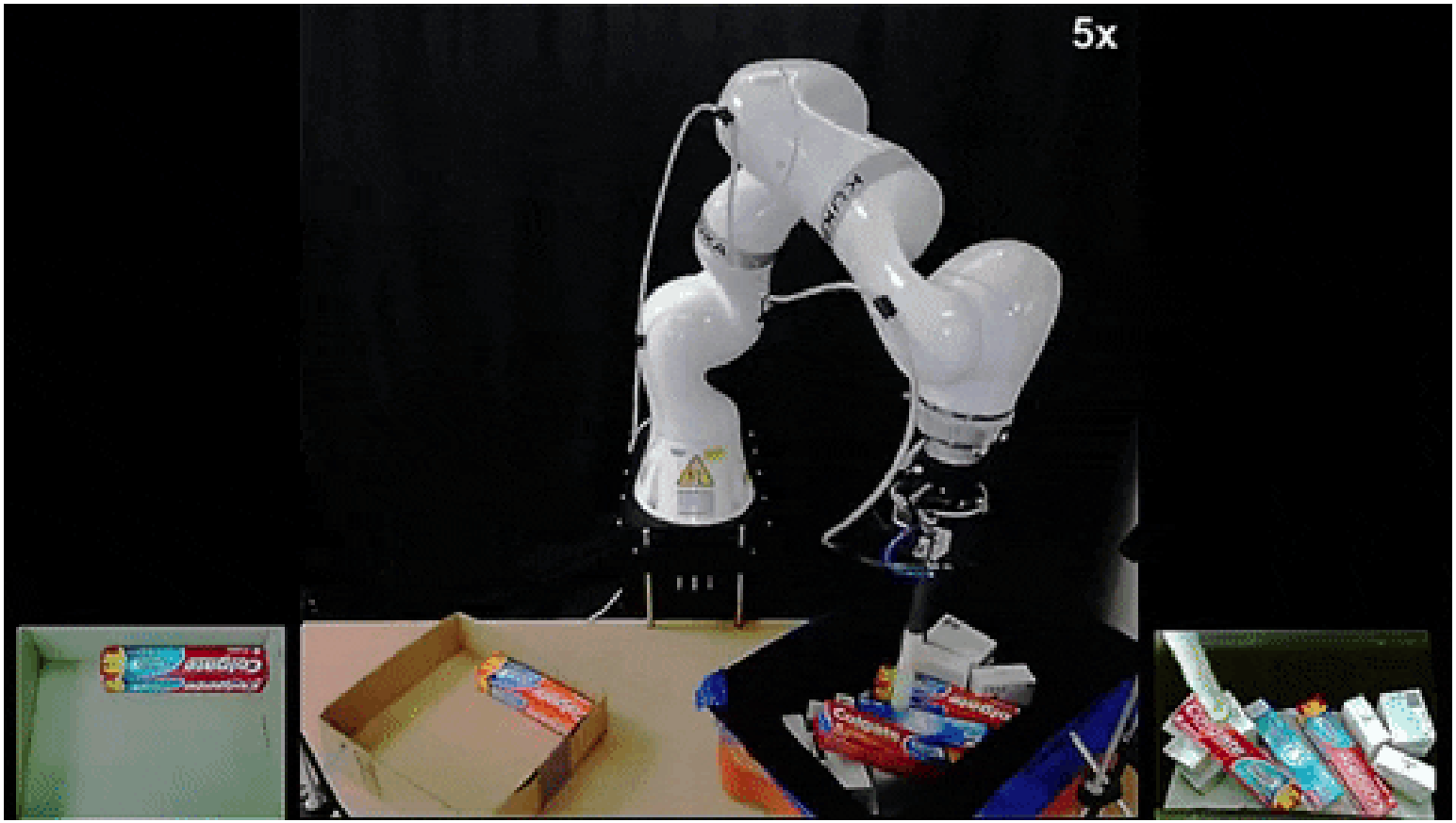
Decision Making AI using Reinforcement Learning

Self Teaching Neural Network when Answers and Patterns are Unknown

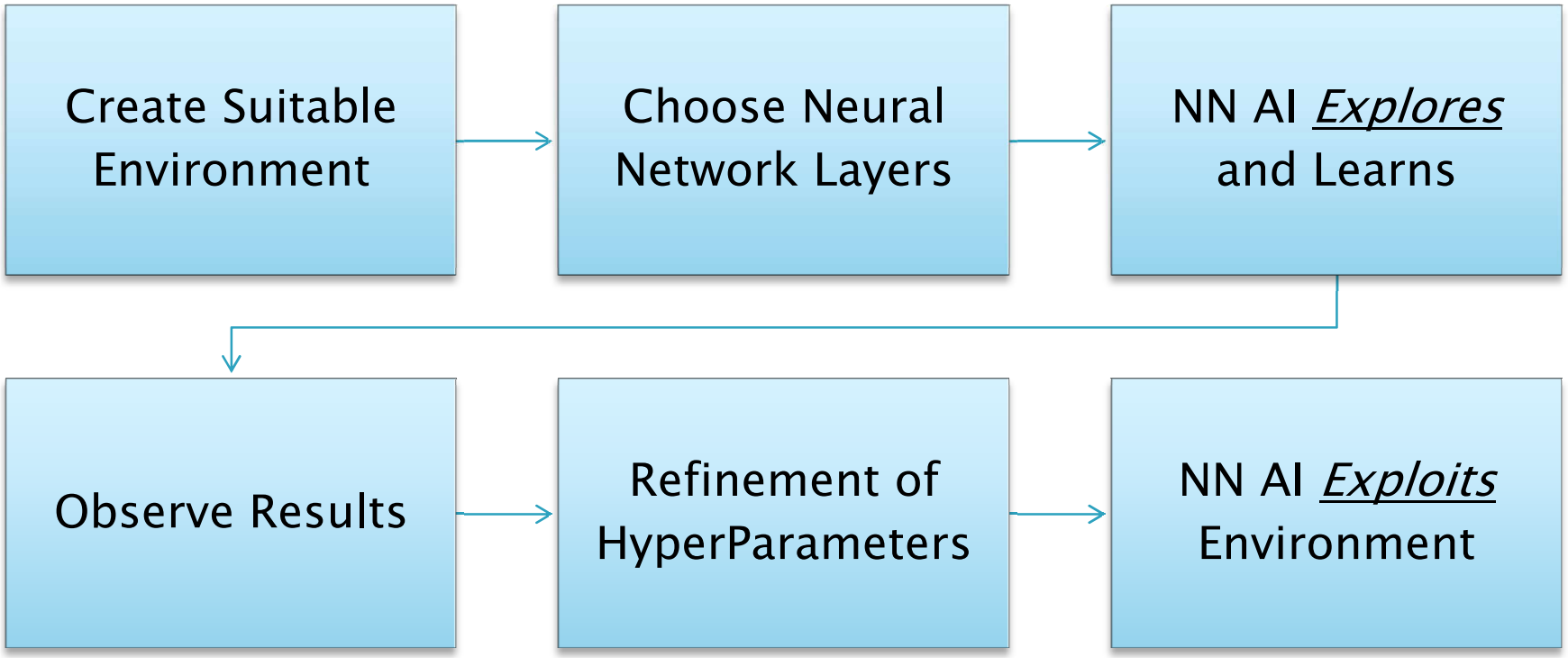
By William Li

Why Reinforcement Learning

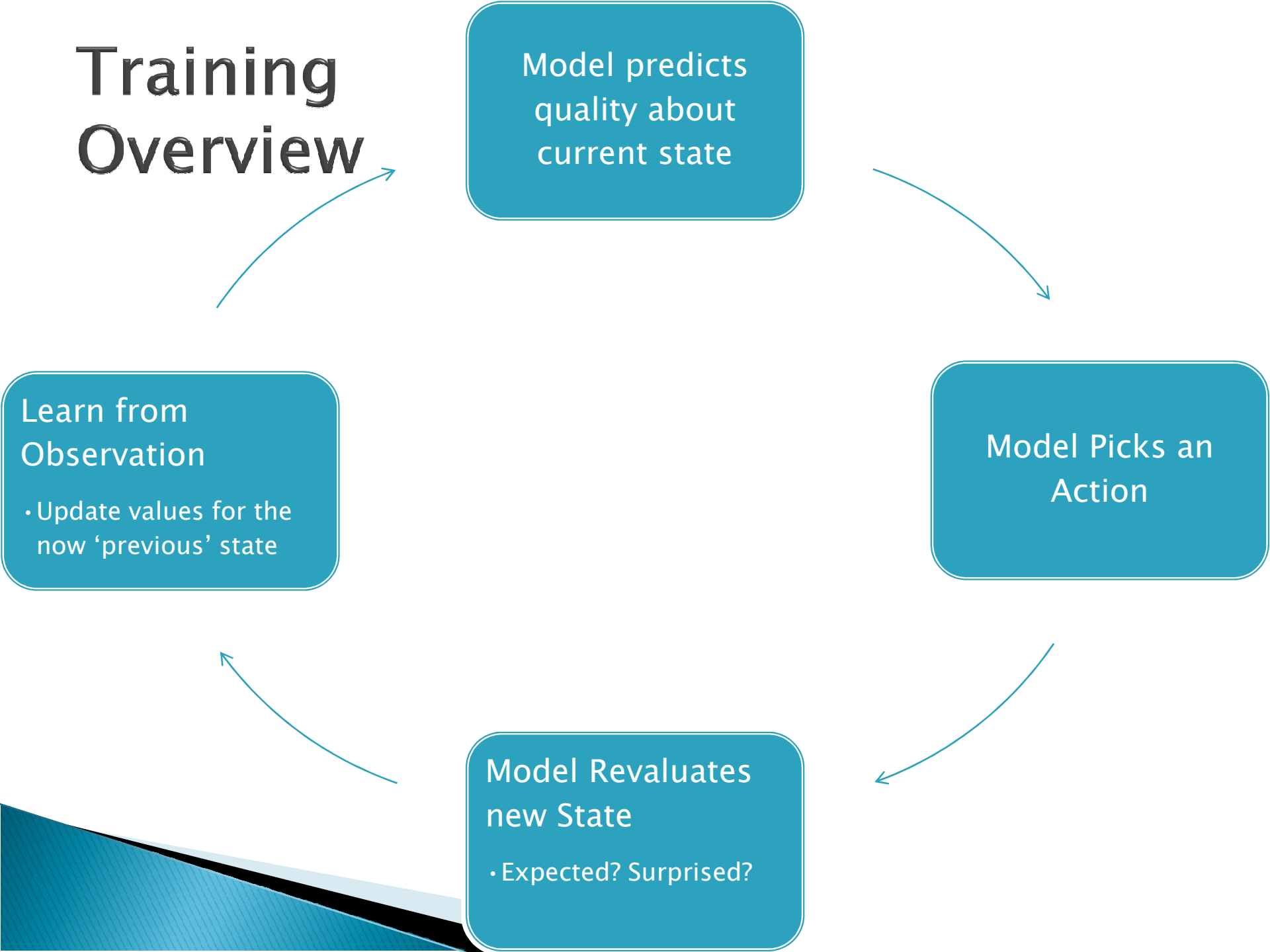


Img Source: See Appendix A

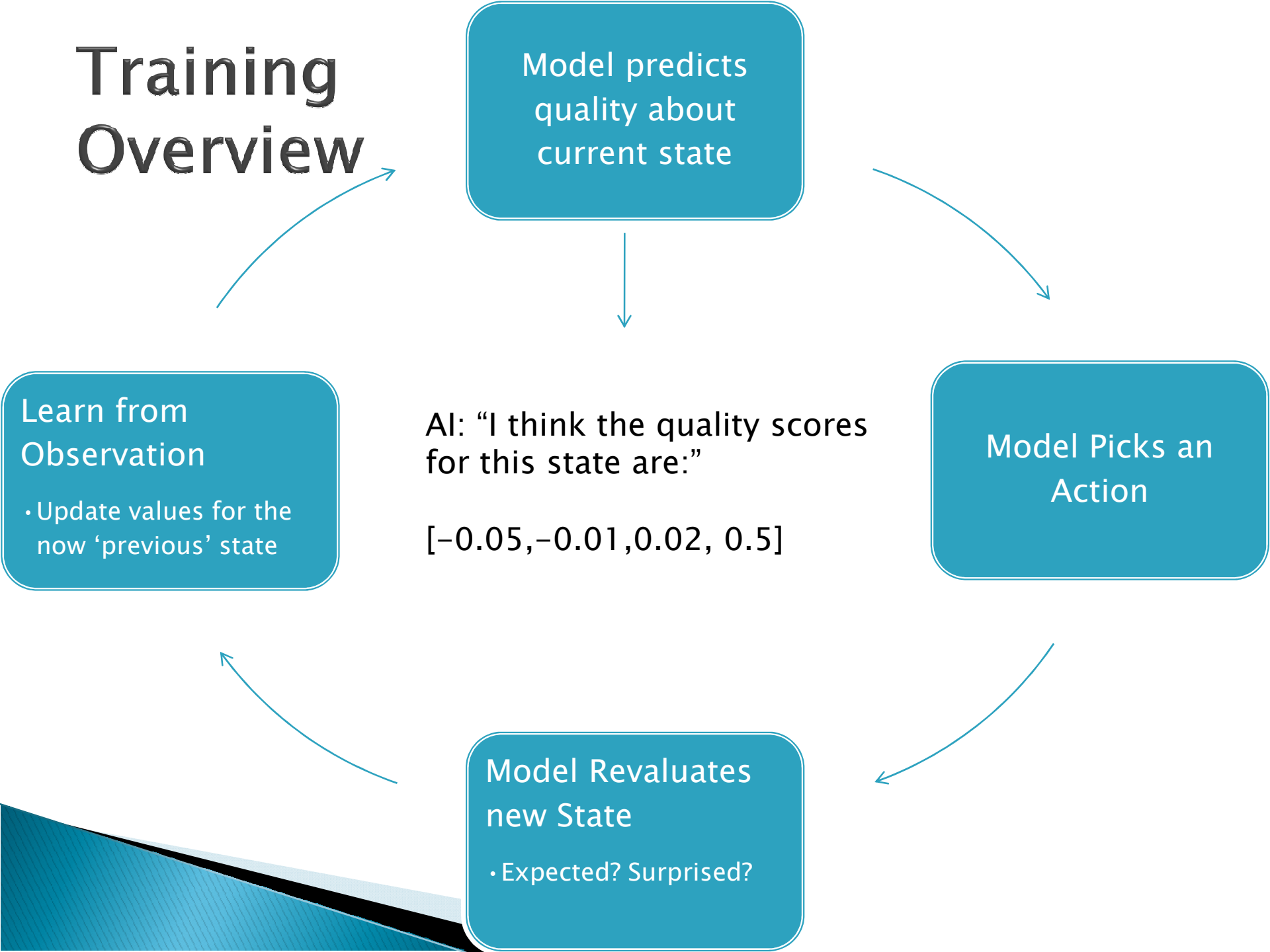
Plan of Attack



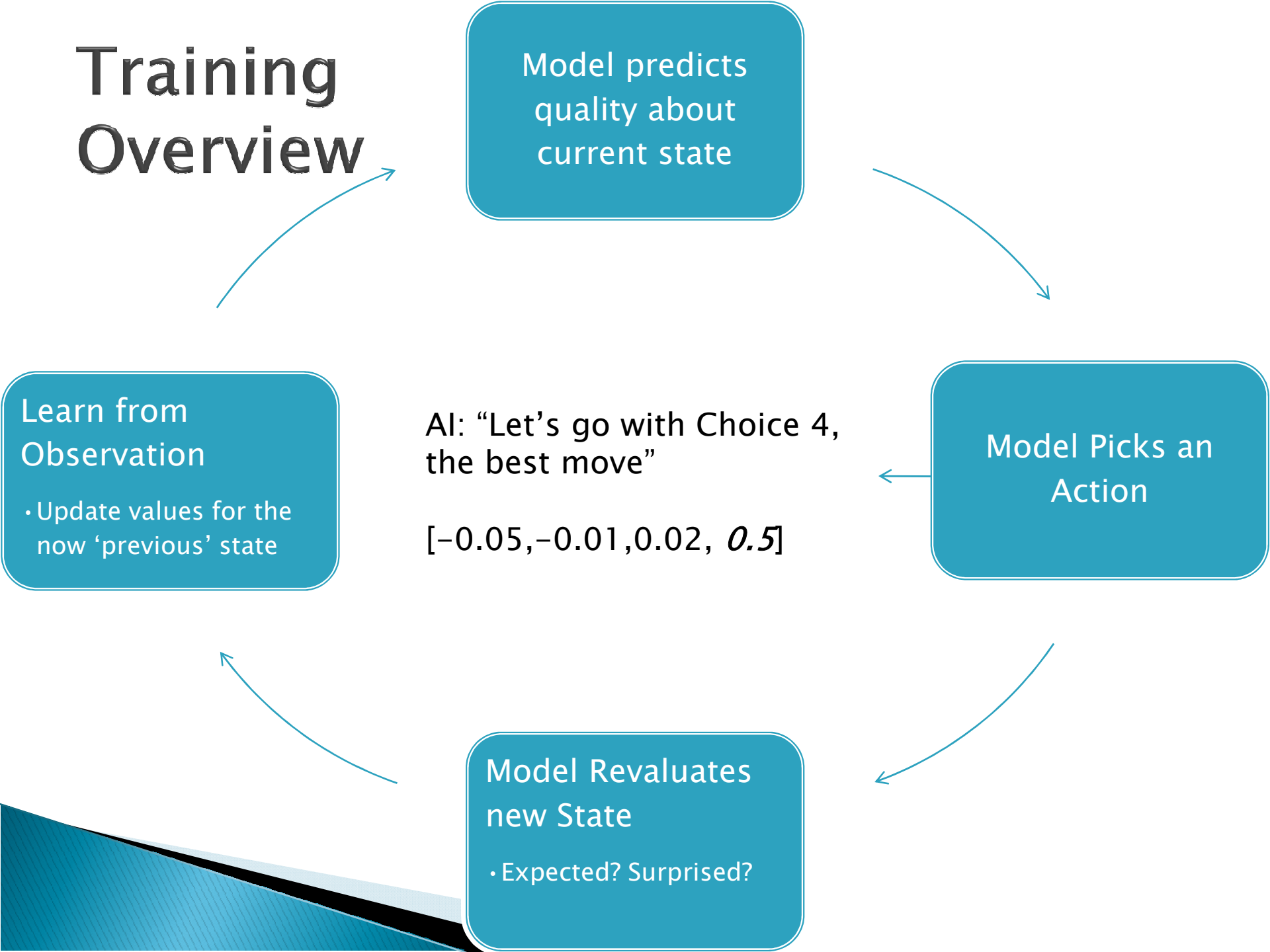
Training Overview



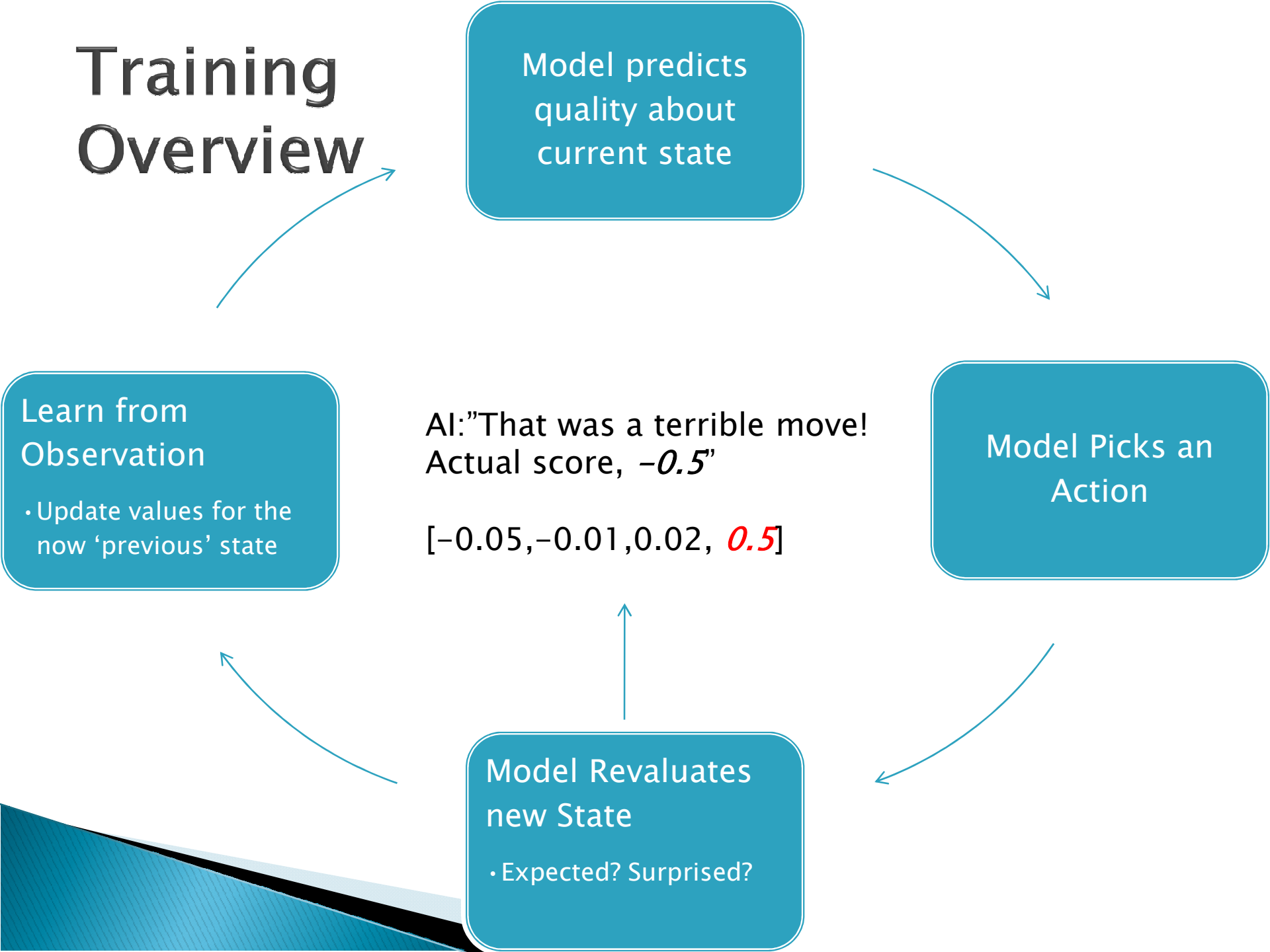
Training Overview



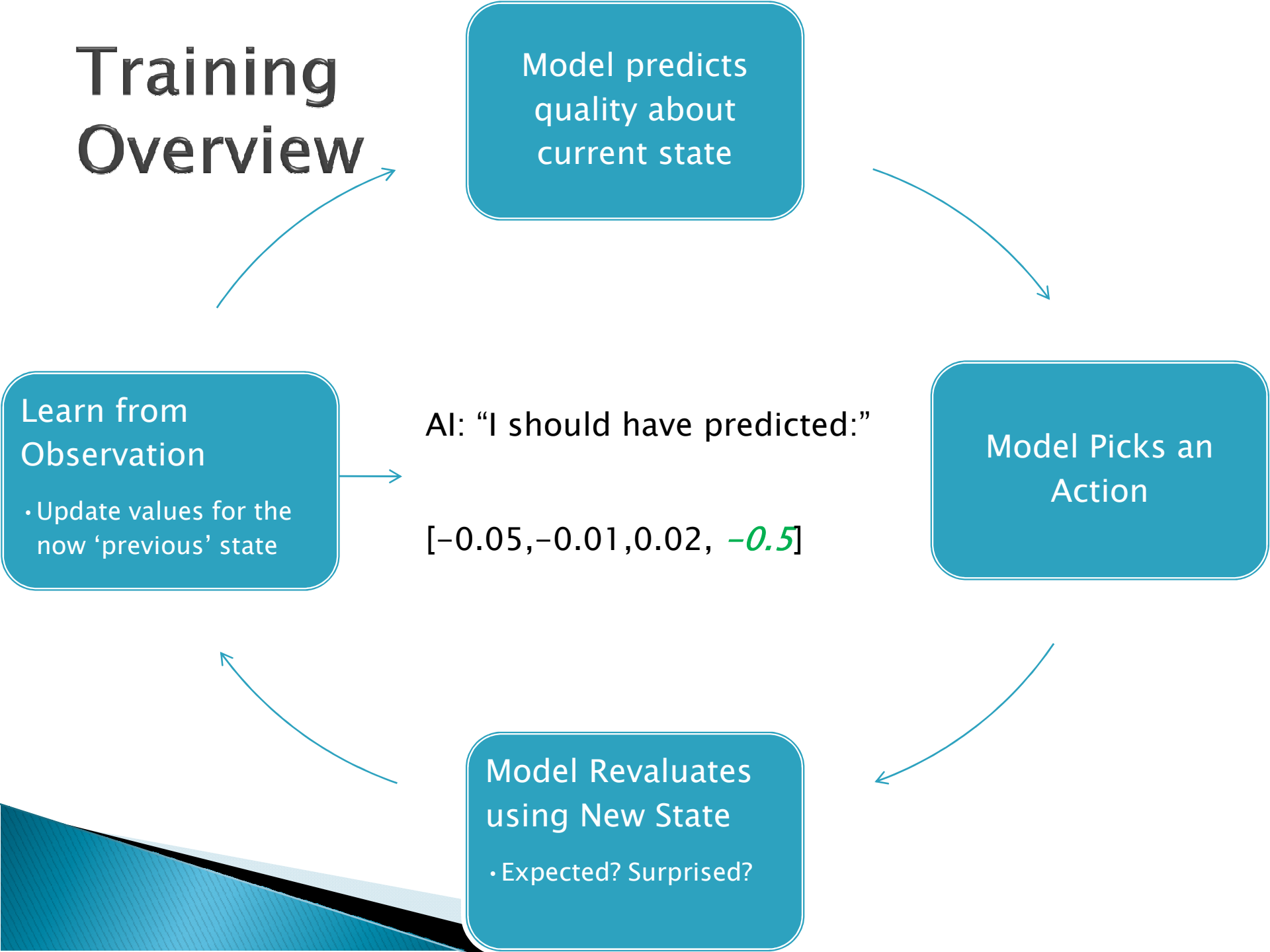
Training Overview



Training Overview



Training Overview



Training Overview

Model predicts quality about current state

Model Picks an Action

Model Revaluates new State

- Expected? Surprised?

Learn from Observation

- Update values for the now 'previous' state

Q-Learning

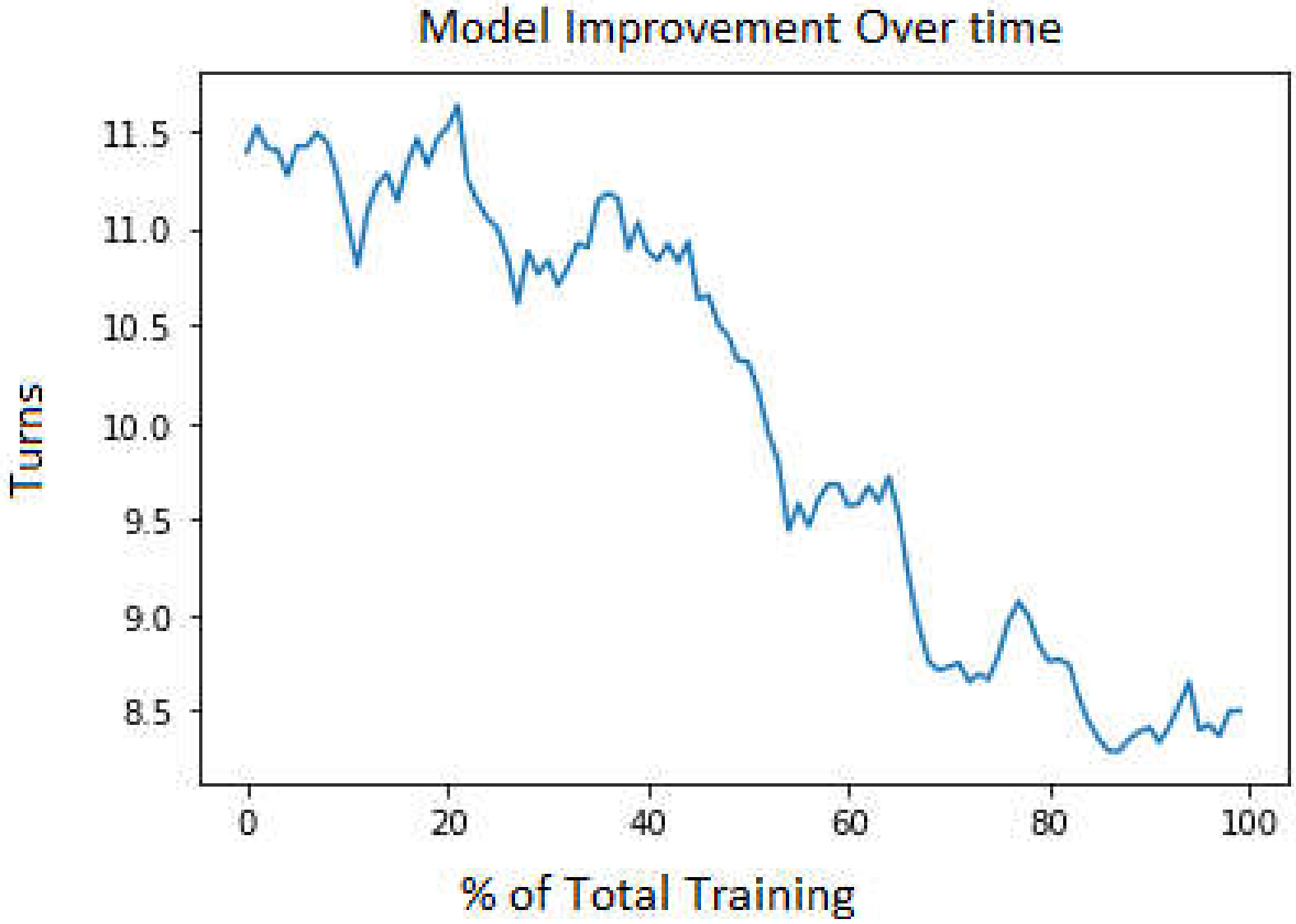
$$Q(s_t, a_t)^{Updated} = r + \gamma \left[max_a Q(s_{t+1}, a) - Q(s_t, a_t)^{old}\right]$$

Key Insight

- Proportionate Reward
- Deeper Brain
- Train backwards from the end (If possible)

- More time!

Results


Model Improvement Over time
Turns
11.5
11.0
10.5
10.0
9.5
9.0
8.5
0
20
40
60
80
100
% of Total Training

Conclusion and Next Steps

Neural Network Model Learned Game!

Improvements from 11.5 turns with random moves, to 8.5 with learned patterns

<u>What's Next</u>

- More Training
- Increase Complexity
  - Add More States
- Introduce Competitive Versus Element

Thank you for Listening
Any Questions? Catch me in Session Room After!

- See the code at
- https://github.com/li-william-wl/ReinforcementLearningAI

- Play the Game the AI trained on at
- https://bit.ly/3ch2GWA

- Appendix Source:
- A: https://medium.com/robotic-arm-control-using-deep-reinforcement/robotic-arm-control-using-deep-reinforcement-learning-3443969f7d0f

Game Rules