Automatic Mechanical Ventilation Control (AMVC) (Team 18 Final Report)

Mentor: Dr Antwan D. Clark
Team Members: Haoyu Shi, Jiahe Xu

***Abstract*— Mechanical ventilators are commonly used to assist the breathing of patients with respiratory diseases. Many mathematical models of mechanical ventilators have been proposed [3][4] to describe lung behavior during artificial ventilation. To describe the behavior of the ventilator and lung, there are parameters proposed in to help researchers understand and manage ventilatory support. With mathematical models of the lung simulating the lung's reaction and ventilator management parameters displaying the lung's condition, a controller could be designed to control the ventilator output signals and adjust the output signals according to the lung's condition shown by the ventilator management parameters. In this paper, we used the mathematical model proposed in [4], and management parameters in [12] to build the environment, and then we used Reinforcement learning methods to train a controller that helps the output of the lung model reach the desired state. The desired state could be a doctor-assigned value or the range of normal people's lung output values. The results show that our trained agents are able achieve a target value or close to target value for a fair number of cases for each input case.**

I. Introduction

Respiratory diseases affect the health of human beings and the quality of life in many ways. The world-range spread of coronavirus 2019 (COVID-19) has taken millions of people's lives. In clinical treatment, patients need ventilators to deliver enough oxygen to keep the body going. Mechanical ventilation aims to maximize the gas exchange to the patient's lung and minimize the ventilator-caused lung injury as possible at the same time. There are two categories of mechanical ventilation (i) pressure-controlled ventilation (PCV); (ii) volume-controlled ventilation (VCV). PCV entirely relies on patient-triggered breaths and the ventilator cycles between two different pressures (PEEP and PIP). PCV often is used for ventilator weaning since it augments patients' breathing efforts but not a fixed tidal volume. The VCV, on the other hand, will try to keep the volume constant through a feedback control and it is commonly used in operation theatres and intensive care units (ICU). PCV and VCV have their pros and cons, doctors need to decide which one to use according to the patients' lung disease and patient lung characteristics. Incorrect settings of mechanical ventilation could cause lung injury [13]. In our work, the lung mechanics are observed through the output parameters of the lung model, and we determined a set of normal ranges for these parameters. In any case, when the output leaves the normal range, there will be a huge cost added to the ventilator controller agent, which forces the outputs back to normal ranges.

In our work, we followed [14] to decide the inputs of PCV and the output of the lung model and followed [14] to compute the ventilator-derived parameters to help manage ventilation. Inputs of PCV are: (i)PEEP (positive end-expiratory pressure); (ii)PIP (peak inspiratory pressure); (iii)RR (respiratory rate); (iv) I/E ratio (inspiratory expiratory ratio). PEEP is the alveolar pressure above the atmospheric pressure at end-expiration. PEEP aims to: keep lungs open at the end of expiration and prevent the opening and closing of distal airways and alveolar units. Normally, PEEP could be set from 5 up to 25 cmH2O [15], in our study we set the range of PEEP as 3 to 15 cmH2O (or the limit of the safe range) since the PIP of a normal ventilator machine can produce is 40 cmH2O. PIP is the peak inspiratory pressure (maximal pressure), the difference between PIP and PEEP is driving pressure (delta P), which determines the tidal volume of the lung. RR is the respiratory rate of the patient; it represents how many times the patient respirate in a minute. To simulate the lung of different patients', we set the range of RR as 12 to 20 times/per-min taking both [16] and [4] into consideration. I/E ratio is the ratio of inspiratory time and expiratory time in each respiratory cycle, we set it as 1/4, 1/3, 1/2, or 3/4 following [17]. According to these input signals, we can get corresponding output signals (Flow, Pressure, Volume) and compute useful parameters such as mechanical energy, mechanical power, airway resistance, and dynamic compliance for ventilation management.

The whole simulation is implemented in Simulink. Followed the procedure in [4], we reproduced the result. The simulation period is set as 60 seconds since the waveforms of flow, pressure, and volume become periodic with less than 3% fluctuation over the value. If the simulation period is longer than 30 seconds, the overall result doesn't change much as well (less than 5% on the value). Besides, we experimented with different initial lung settings (volume, pressure) and the output signals all come to the same results.

II. Related Work

In previous work [3][4], multi-component mathematical lung models have been proposed. Their results have been proved to be close to the real lung reaction, and [5] added more lung's muscle feedback into consideration. [14] listed all the important parameters that could be used in ventilation management. For now, all the settings of mechanical ventilator have to be manually monitored and set according to doctors' and nurses' experiences, which demands lots of man power and increases many uncertainties in the treatment process. Bases on previous works, we built up a lung model and a ventilator system in simulation environment. With proper reward functions, DRL methods such as [8][9] could be used to train a controller agent to shorten the time for patients to reach a safe lung condition.

III. Theoretical Development & Analysis

We carried out our experiment by first implementing a mathematical model that simulates both the Pressure Controlled Ventilator(PCV) signals and the patients' lung responses based on Al Nagger's paper.[4] The model allows us to simulate the patient's response under various inputs. Then, using the data we gained from the simulation environment, we used R-peak calculations to get the peaks and bottoms of the waves of the outputs Q, P and V.[11] The peak data were next used to devise MDP states and actions. Finally, using the MDPs as basis, we wrote our custom AI-GYM environments and trained our RL agents. All steps will now be covered in more detail in the following paragraphs.

A. Mathematical Model

The mathematical model for the mechanical ventilation system in our work includes the pressure signal from the PCV device and the mathematical lung model, we will describe them in this section. The models were built using the methods described in [**?**]

- Pressure signal of PCV
  The pressure signal should represent the breathing behavior and activities. In our work we choose a typical waveform of pressure signal which includes: pressure rising time(τ),Peak inspiratory pressure(PIP),positive end-expiratory pressure($PEEP$), respiratory rate(RR), inspiratory time(T_{in}). RR and T_{in} determine the period of a respiratory cycle and the length of inspiratory time. In each respiratory cycle the PCV pressure signal will be as in equation 1.

$$P(t) = \begin{cases} (PIP - PEEP)\dfrac{t}{\tau} + PEEP, \ 0 \le t \le \tau \\ \qquad\qquad\qquad PIP, \ \tau \le t \le T_{in} \\ \qquad\qquad\qquad PEEP, \ T_{in} \le t \le T_{ex} \end{cases} \tag{1}$$

- The Mathematical lung model
  The suggested mathematical lung model is a multi-compartment model. This model was proposed in [19].This model has been converted to the mathematical model using the following equation(in frequence domain). The output of the system represents the airflow changes, which could be used to calculate the airflow by a integral operation over time. It is shown in equation 2 with the following variables included:
  - R_C:the airflow resistance of the central airways
  - R_P:the airflow resistance of the peripheral airways
  - C_L:the capacity of the alveoli
  - C_W:the capacity of the chest wall
  - C_S: a shunt capacitance known as "dead space" of air

RR(60/TCT)	IE(%, constant)	PIP	3	4	5	6	7	8	9	10	11	12	13	14	15
15	50	20	113.307	106.643	99.9768	93.3207	86.6586	79.9225	73.3267	66.6608	59.9947	53.3287	46.6319	39.9723	33.3121
15	50	21	119.97	113.308	106.642	99.9752	93.3246	86.5814	79.9243	73.3268	66.6608	59.9947	53.3287	46.6621	39.973
15	50	22	126.655	119.97	**113.308**	**106.642**	**99.9752**	**93.3246**	**86.5814**	**79.9243**	**73.3268**	**66.6608**	**59.9947**	**53.3287**	**46.6621**
15	50	23	133.291	126.655	119.97	**113.308**	**106.642**	**99.9752**	**93.3246**	**86.5814**	**79.9243**	**73.3268**	**66.6608**	**59.9947**	**53.3287**
15	50	24	139.974	133.291	126.655	119.97	**113.308**	**106.642**	**99.9752**	**93.3246**	**86.5814**	**79.9243**	**73.3268**	**66.6608**	**59.9947**
15	50	25	146.653	139.974	133.291	126.655	119.97	**113.308**	**106.642**	**99.9752**	**93.3246**	**86.5814**	**79.9243**	**73.3268**	**66.6608**
15	50	26	153.319	146.653	139.974	133.291	126.655	119.97	**113.308**	**106.642**	**99.9752**	**93.3246**	**86.5814**	**79.9243**	**73.3268**
15	50	27	159.985	153.319	146.653	139.974	133.291	126.655	119.97	**113.308**	**106.642**	**99.9752**	**93.3246**	**86.5814**	**79.9243**
15	50	28	166.619	159.985	153.319	146.653	139.974	133.291	126.655	119.97	**113.308**	**106.642**	**99.9752**	**93.3246**	**86.5814**
15	50	29	173.284	166.619	159.985	153.319	146.653	139.974	133.291	126.655	119.97	**113.308**	**106.642**	**99.9752**	**93.3246**
15	50	30	179.948	173.284	166.619	159.985	153.319	146.653	139.974	133.291	126.655	119.97	**113.308**	**106.642**	**99.9752**
15	50	31	186.613	179.948	173.284	166.619	159.985	153.319	146.653	139.974	133.291	126.655	119.97	**113.308**	**106.642**
15	50	32	193.277	186.613	179.948	173.284	166.619	159.985	153.319	146.653	139.974	133.291	126.655	119.97	**113.308**
15	50	33	199.981	193.277	186.613	179.948	173.284	166.619	159.985	153.319	146.653	139.974	133.291	126.655	119.97
15	50	34	206.647	199.981	193.277	186.613	179.948	173.284	166.619	159.985	153.319	146.653	139.974	133.291	126.655
15	50	35	213.27	206.647	199.981	193.277	186.613	179.948	173.284	166.619	159.985	153.319	146.653	139.974	133.291
15	50	36	219.985	213.27	206.647	199.981	193.277	186.613	179.948	173.284	166.619	159.985	153.319	146.653	139.974
15	50	37	226.647	219.985	213.27	206.647	199.981	193.277	186.613	179.948	173.284	166.619	159.985	153.319	146.653
15	50	38	233.13	226.647	219.985	213.27	206.647	199.981	193.277	186.613	179.948	173.284	166.619	159.985	153.319
15	50	39	239.975	233.13	226.647	219.985	213.27	206.647	199.981	193.277	186.613	179.948	173.284	166.619	159.985
15	50	40	246.643	239.975	233.13	226.647	219.985	213.27	206.647	199.981	193.277	186.613	179.948	173.284	166.619

Fig. 1. Flow Values

  - C_T: total compliance of airways

  These values are the settings of lung models. For different lung models, we need to change these parameters and run the simulation to get the output of lung model.

$$\frac{I(s)}{P(s)} = \frac{s^2 + \left(\frac{s}{R_P \times C_T}\right)}{R_C \times s^2 + \left(\frac{1}{C_S} + \frac{R_C}{R_P \times C_S}\right)s + \left(\frac{1}{R_P \times C_S}\right)\left(\frac{1}{C_L} + \frac{1}{C_W}\right)} \tag{2}$$

B. Markov Decision Process(MDP)

From the data, we extracted the ranges of the corresponding output to each set of input parameters. They were later used as each individual state of our MDPs.

- MDP for Flow:
  Because we found that for each set of input parameters, the max and min values have the same absolute value, therefore, we are using the max absolute value of the flow data as the representation of each state. The actions in the transition graph is + 1, - 1 or maintain(+ 0). Several boundaries were included in our MDP, anything out of bounds are considered warning states where an action must be made to revert that. They are as followings:
  - $PEEP \in [3, 15]$
  - $PIP \in [20, 40]$
  - $IP \in [9, 18]$

  The first two variables' range has already been stated in the Introduction, the third constraint was added because we found that the ideal range of Flow Rate is 60 to 120 [18]. After running our simulation model, we output the resulting data in a table shown in figure 1 on page 2 which consists of the range of IP that would allow the flow to stay in range. The orange part is the legit range.

  An example of the MDP is shown in figure 2 on page 3, this is the case for $PIP = 25$ and we have an MDP for all PIPs that are in the orange section, which means that the PIPs within range would allow at least 1 valid state in the structure.The red states are warning states while the grey states are allowed states for the ventilator.

- MDP for Pressure and Volume Because we did not find certain bounding values for the range in which the volume itself should be in compared to the case of MDP for Q, the warning states are only for the input values in which the PEEP and PIP are out of range. The PEEP and PIP ranges are still the same as the MDP for Q above. However, we divided the MDP for V into two,

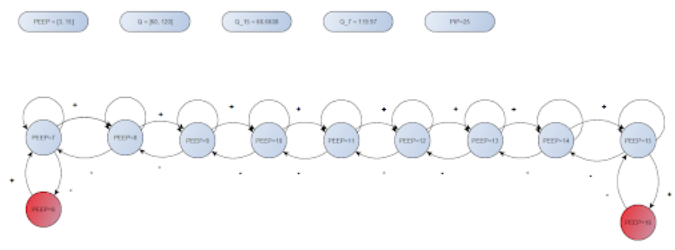

Fig. 2. Example MDP for PIP = 25

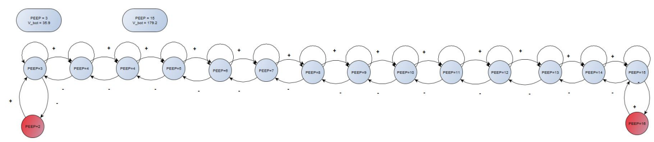

Fig. 3. MDP for Bottom Values of Volume

one in which the states represent the maximum values of the volume while the other represents minimum values of the volume. The MDP for P was not individually listed as the range of the Pressure is a direct reflection of the PIP(maximum) and PEEP values(minimum). The full MDP for the minimum values of V is in figure 3 on page 3 while the one for the maximum values is in figure 4 on page 3.

- MDP for Time When developing the RL networks, we found that we needed an agent that is able to adjust the time periods and inspiratory/expiratory times. Therefore, this MDP was added and is based on values of RR and I/E ratio. The boundaries for this MDP is $RR \in [12, 20]$. An example with an I/E ratio of 1:3(where inspiratory time is 25% of total cycle time) is shown in figure 5 on page 3 where the warning state is when the RR value is out of range.

C. Reinforcement Learning

1) AI-GYM environments Two custom AI-GYM environments were built based on MDP for Q and MDP for Time. One is used to train an RL agent that can adjust PIP and PEEP parameters while the other is trained to make adjustments for RR and I/E Ratio.

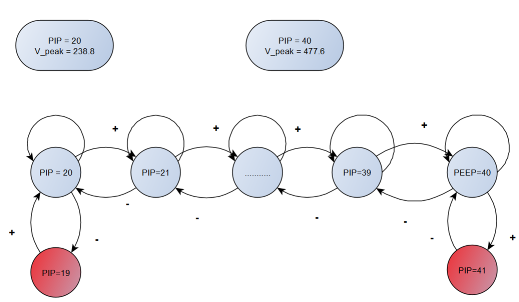

Fig. 4. MDP for Peak Values of Volume

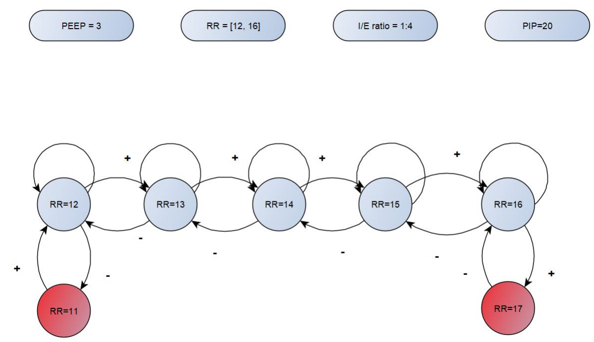

Fig. 5. MDP for I/E ratio = 1 : 3

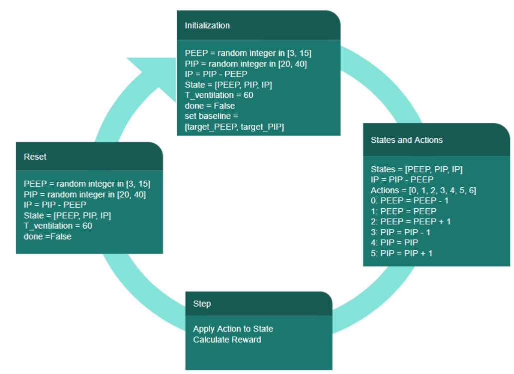

Fig. 6. AI-GYM Environment for Q

The AI-GYM environments were composed partially by learning from [7]

- Environment for Q:
  The main structure of the environment for flow is shown in figure 6 on page 3 The reward is calculated depending on its distance from the target value. For PEEP, there is no exact value, a value is provided so that it won't be too close to edge values while it is encouraged to get to the exact target PIP value. If any of the values are in a warning state described in the MDP section, the reward is always -80. For the IP value, as long as it is within range of [9, 18], the reward is 20. While on the other hand, PIP and PEEP have varying rewards.
  - PIP:
    if the current PIP is equal to the target, a reward of 40 is given. Otherwise, the calculation function is equation 3

$$Reward(PIP) = 20 - |PIP - PIP_{tgt}| \quad (3)$$

  - PEEP:
    PEEP's reward is PIP's calculation equation without the target value reward. It is shown in

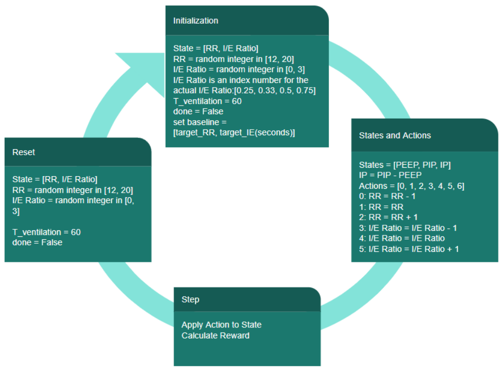

Fig. 7. AI-GYM Environment for T

equation 4

$$Reward(PEEP) = 20 - |PEEP - PEEP_{tgt}| \quad (4)$$

- Environment for RR and I/E Ratio:
  This environment is similar to the flow environment in that it pulls the agent towards the target. The main structure of the environment for flow is shown in figure 7 on page 4 The input value is three values: RR, time in seconds and a boolean. The last boolean decides whether the given time is the exact time for inspiratory time or expiratory time, True for inspiratory and False for expiratory. The greatest difference compared to the previous environment is that it has two target values and reaching both will automatically stop the current episode and grant a completion reward of 1000. Aside from the general rule that any out of range warning states are given a penalty of -1000 reward, the rewards for I/E ratio and RR is as followings:
  - RR:
    If the value is within range, the reward is determined by how close or if it is at the target. If it is the exact target value, the reward is 90. Otherwise, the calculation function is through equation 5

    $$Reward(RR) = 3 \cdot (20 - |RR - RR_{tgt}|) \quad (5)$$

  - I/E Ratio:
    If the value is within range, the reward is determined by how close or if it is at the target. If it is the exact target value, the reward is 60. Otherwise, the calculation function is through equation 6

    $$Reward(IE) = 30 - |IE - IE_{tgt}| \quad (6)$$

    However, the listed IE is not a fixed value, it is the best IE ratio that can be chosen, the inspiratory/expiratory time that is closest to the input target time. For example, if the target time is 2 seconds in inspiratory time but the current RR is 12, no IE ratio offers exactly the same time. The best result will now be the target which is 0.5, 2 in index.

2) DeepQLearning(DQN)
   The Reinforcement Learning(RL) method we would like to use is Q learning. We are using a discount rate γ of 0.99. The calculation of Q value is equal to equation 7 The prediction network will be used to predict what the reward could be.

   $$reward + next\_state\_values \cdot \gamma \quad (7)$$

   However, the difference between two states is very small, making it very hard for the network to make a decision and calculate the loss. Therefore, a prediction and a target network is implemented.[8] We use the difference between the target and the prediction network to produce a loss and learn the gradients. The target network is an exact copy of the structure of the prediction network, During initialization, the parameters are copied so that they are identical. The parameters of the target network is updated only after certain episodes. We will then move on to our implementation which was modified and based on [10]
   - Network Structure: a linear network with two hidden layers, each with 256 parameters, the input is the dimension of the state which is 3 in this case and the number of actions as output, 6 in the current case.
   - Update interval for target network: 10 episodes
   - Loss Function: Mean Squared Error(MSELoss)
   - Optimizer: Adam
   - Hyper-Parameters:
     - Learning Rate: 0.001
     - Episodes:300
     - Batch size: 128
     - Epsilons:
       Epsilon values dictate how much exploration and exploitation the agent is going to have during the process. In exploration phase, the agent makes a completely random move and when it's exploiting, it makes a move based on the prediction network.
       * Epsilon Start = 1.0
       * Epsilon End = 0.01
       * Epsilon Decay = 5e-4
       * Memory Size = 10000

IV. Empirical Results & Simulation

In this section you want to present your methodology for validation. This can either via data or via simulation. If you're using data, please first describe the dataset used. Next, describe how you're validating the proposed analyses (in Section 3) into the dataset. If you're simulating data, please first describe the methodology used then afterwards, describe how you're validating the theoretical analyses and simulations.

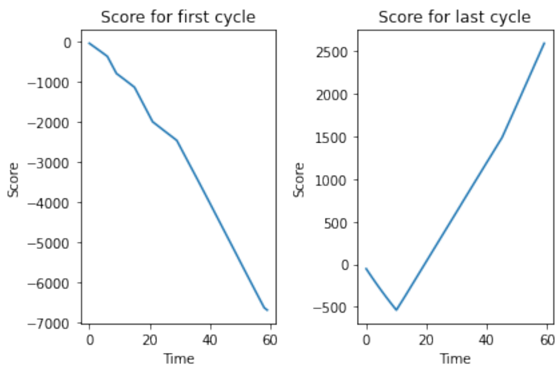

Fig. 8. Cumulative Score of first and Last episode for RL_Q

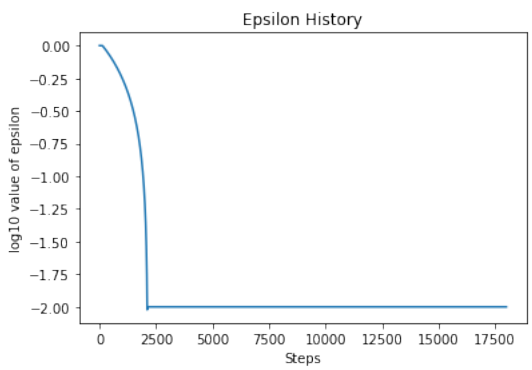

Fig. 9. Epsilon History for RL_Q

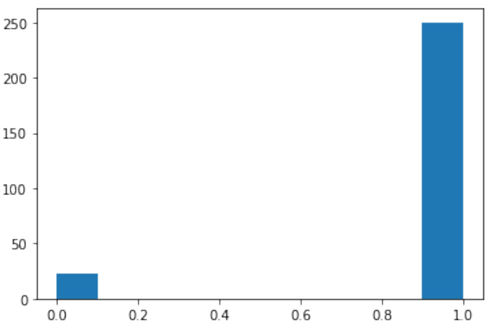

Fig. 10. Difference between target and test PIP result for RL_Q

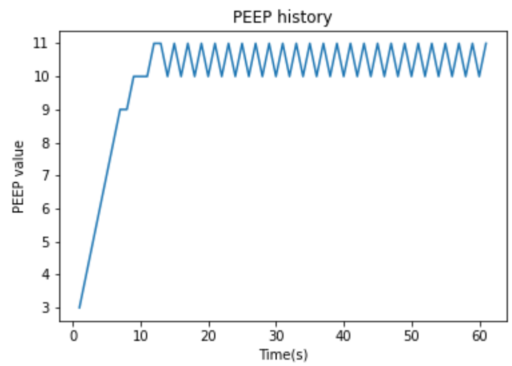

Fig. 11. PEEP history for RL_Q test case

A. Results for RL_Q

Due to time issues, completion and optimization of the Agent was completed just before end of semester and therefore, the number of tests currently are still few. I would like to present data of one set of target case.
The case is when the target PIP is 25 and PEEP is maintained around 10. Figure 8 on page 5 shows a comparison between the cumulative score of the first and the last episode. The graphs shows that the loss went from going completely downwards during the first cycle to going continuously upwards after a short while. This shows that the agent is slowly learning to avoid penalties. The epsilon history is shown in figure 9. In order to make the trend more clear, the epsilon values have been applied a base 10 log.

During the tests for this case, we reset the environment to all valid combinations of PEEP and PIP then let the agent adjust the parameters on its own. All results showed no sign of the agent getting a negative reward which means that it avoided all warning states. In addition, figure 10 on page 5 is a histogram that shows the absolute value of the difference between the PIP value of the last state of each test case and the target PIP. The numbers are 23:250 for the number of cases with differences of 0 and 1 correspondingly. This shows that despite the fact that the target value cannot be always reached, there is a high possibility of finding it in close proximity. The figures 11, 14, 12 and 13 are graphs that show the history of the environment states including all input values PEEP, IP, PIP and also the rewards of each step during one test cycle where the starting state of the environment is $PEEP = 15$ and $PIP = 20$. As can be seen from the figure 12, the PIP value rose slowly towards 24, which just 1 value below 25. The PEEP in figure 11 rose and fell during the process in order to keep the resulting IP still in range while PIP is adjusted. The horizontal area is where adjustments are made back and forth between two values so that the IP can still be balanced within range. This can be directly seen from figure 14, where the IP is always maintained within range. The oscillations in the figure 11 is mainly due to the fact that there is no target for the PEEP, the rewards around the area are all similar and the agent can't decide which state to choose.

B. Results for RL_T

In the Reinforcement Agent trained for time, the results are as following. The instructions for the agent is to reach a target RR and target inspiratory/expiratory time in seconds. The requirement is for the time in seconds to be maintained close to the input during the process.
The example set of figures shown is for an input of $[15, 2, True]$, this means that the training target is an RR of 15 and an inspiratory time of 2 seconds. As can be seen from figure 16, the epsilon here looks prolonged compared to the previous training process of RL agent for Q. This is because if both targets are met, the environment stops and grants a done reward. The training epochs is still set to 300 but the

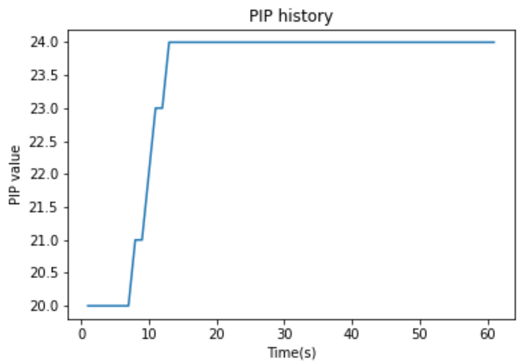

Fig. 12. PIP history for RL_Q test case

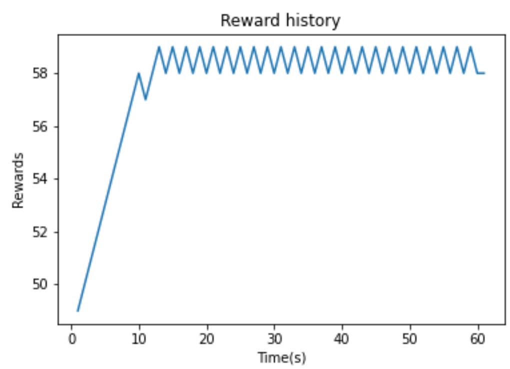

Fig. 13. Reward history for RL_Q test case

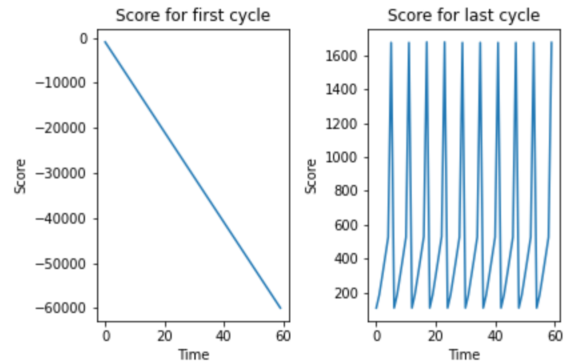

Fig. 15. Cumulative Score of First and Last 60 steps for RL_T

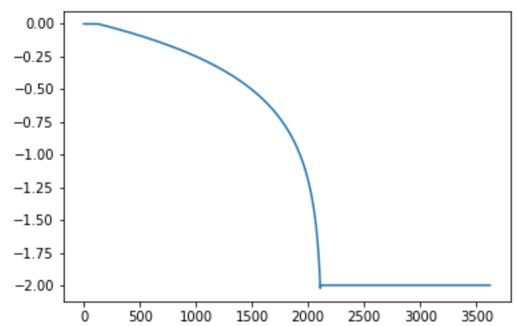

Fig. 16. Epsilon History for RL_T

actual steps in total could be less than $60 \cdot 300$. Figure 15 shows the difference between the first 60 and last 60 steps. The oscillation does not mean that the scores are low. As can be seen, they are drops to the initial state rewards. This means that this not one episode but multiple episodes, the rise periods are when the episode reached the done state, the target values. And the drops are the reset environment. The histograms in figure 18 and figure 17 correspondingly show the difference between the target IE, RR and the test IE(in the form of index), RR results. It can be seen that the agent's ability to adjust to the IE results but still face some difference in RR results up to 3. The positives are that most of the differences in RR lie in the range of 0 to 1. An example of a successful operation by the agent is the test case which starts at the state of $RR = 12$ and $IE = 0$, the target IE index is 2. The process can be seen similar to the Q agent. Both values have adjustments during different times, and in the process, the agent did not raise either value directly to the target but instead adjusted both, ensuring that the time of inspiration stays close to the required target. This testing set was ended early because it reached both targets.

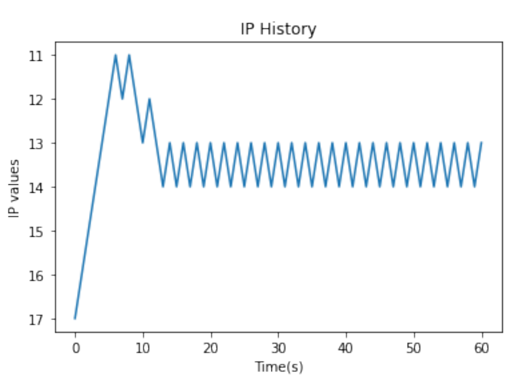

Fig. 14. IP history for RL_Q test case

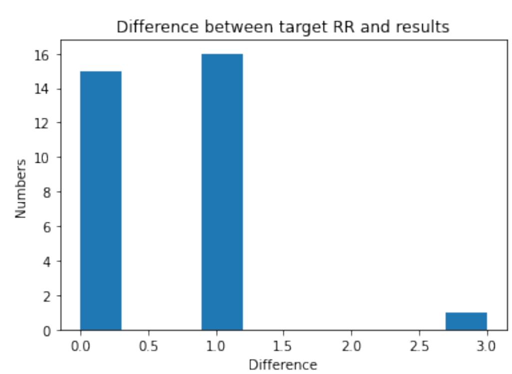

Fig. 17. Difference between target and test RR result for RL_T

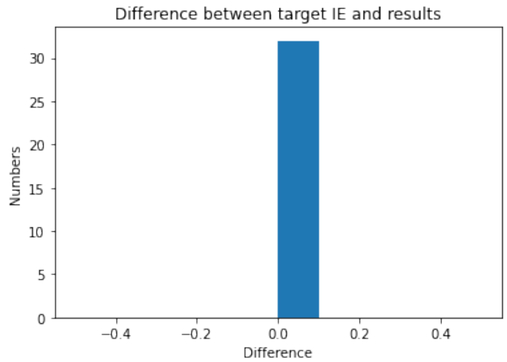

Fig. 18. Difference between target and test IE result for RL_T

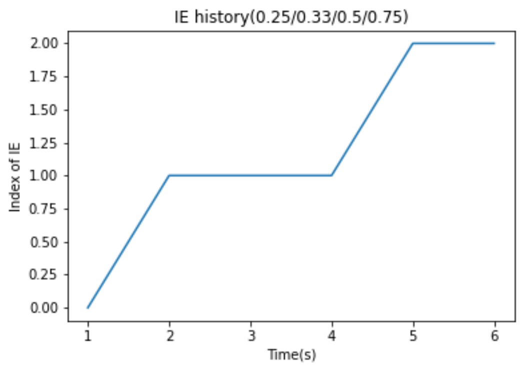

Fig. 19. IE history for RL_T test case

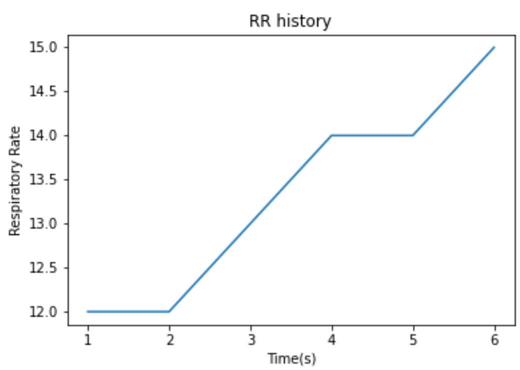

Fig. 20. RR history for RL_T test case

Fig. 21. Reward history for RL_T test case

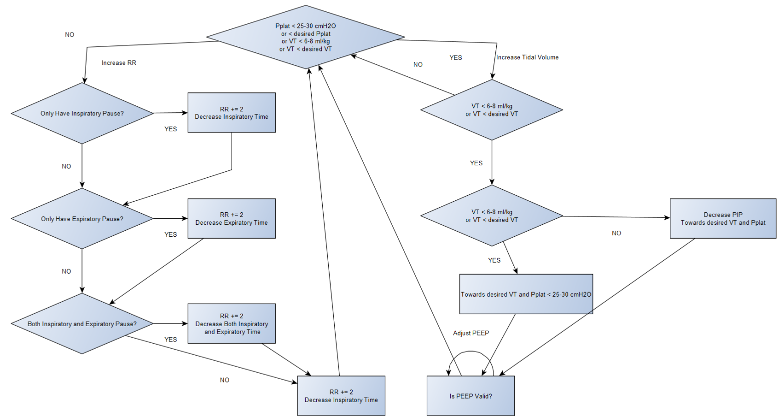

Fig. 22. Structure of Boss Agent

V. Next Steps

Because of the limited time, we have not been able to make optimizations that are crucial to the results. Therefore, we would like to make the following improvements if time permits.

- Adjust hyper-parameters in order for the network to perform better. Because pf the amount of work required is fit in a The current results and graphs show that the network is able to let the agent learn but not to a perfect state.
- A Managing Agent: We have done research and sketches of a Boss agent used to put both sub-agents that we have constructed together, the agent that will give instructions such as target values to each agent. The sketch as shown in figure 22 has already been drawn and if time permitting we plan to have some results before final session. The flow chart for the rules based agent is drawn based on the thesis provided in [2]. The places in the chart where adjustments are needed for RR, IE, PEEP or PIP will be where the two trained sub-agents are used.

VI. Conclusions & Future Work

We have been able to develop a DQN structure for both the Flow and Time period AI-GYM environment. The MDPs in our early stage have been recreated and verified in the custom AI-GYM environments. Although a wide range of tests were not completed at the current stage, we have proof from several cases shown that the agent is able to learn the given targets whether they are derived or directly given. The results although not perfect, the majority of test cases are either exactly on target or very close and with a small error. In the future, we would like to be able to finish the entire structure containing the managing agent, combining all three agents together and running a wide variety of tests so that the DQN networks can be optimized and adjusted to a higher accuracy and more results to prove that. In addition, it is essential that real world data is applied instead just using simulation data that has little to no noise.

VII. APPENDIX

A. Management

1) Contributions The tasks are divided as followings:
   - Collaborated:
     - Paper Reading and Notes
     - Early Stage Selection of Paper
     - Creating (MDPs) for the values Flow(Q), Pressure(P) and Volume(V)
     - Collecting Data from the simulation environment(Mathematical Model of Pressure Controlled Ventilator with Lung)
     - Analyze the data collected from simulation and its waveforms
     - Use the analyzed data to create Markov Decision Processes(MDPs) for the Reinforcement Learning(RL) Agent.
     - Testing and Improving the Network
   - Haoyu Shi
     - Creating AI-GYM environment for the MDPs created
     - Implementing Network Structure for the Deep Q learning network
   - Jiahe Xu
     - Recreation of Mathematical Model of Pressure Controlled Ventilator with Lung(from the Al Naggar Paper)
     - Collected Data using a variety of input parameters for the recreated model in order to produce enough training data for the RL agent.
2) Planned and Accomplished:
   - Our expected deliverables in the first set of slides presented(Minimum and Expected) included the following points:
     - Build a Reliable Mathematical Model(MM) which could simulate the pressure controlled ventilation process and produce lung responses
     - Train an RL agent that is able to make decisions to adjust parameters
     - Initial Proof of concept for Markov Decision Process(MDP) in ventilation controller
     - Documented Code and Performance Metrics
   - Up till now, at the end of the semester, we were able to accomplish most of these. We will go over them point by point.
     - Build a Reliable MM for simulation:
       This is the first step we have implemented during the early stages in order to collect data for both later training and creating the MDPs for the three output values Q, P and V. This was completed in March and the results were compared with the original papers' and also during our weekly meetings, were shown to Hopkins Professors for verification before the next steps were executed. This was completed in early March.
     - Train an RL agent that is able to adjust parameters automatically:
       This step was done towards the end of the project. We were able to build a Deep Q Learning network with a prediction and target network. The structure proved to be useful and satisfactory under some conditions, reaching the set target or range. However, the hyper-parameters might require further optimization.
     - Initial Proof of Concept for MDP in ventilation controller:
       This was done in mid to late March. We devised the MDP according to data we collected from the simulation environment.
     - Documented Code and Performance Metrics:
       Our code and accompanying manuals are all in a private repository where the mentor could fork and save even after the end of the course. Comments are added so that it can be fixed and adjusted in the future. A simple performance metric was used in the results discussion, mainly from the set rewards of the AI-GYM environment. We determined whether it reached the exact desired number, if not, how far away it is.
3) Future Works: Although both of us are graduating and cannot make further changes, we would plan to have the following changes either for the Mentor or in the two weeks before the poster session:
   - Refine hyper-parameters of the network structure including learning rate, batch size etc, to improve the results of the existing agents
   - Assemble together the agents into the proposed hierarchical structure
   - Add in noise or apply real world data to test and train the network
   - Improve the evaluation metrics so that more factors could be added into consideration
4) Gains During the one-semester process, we learned from both the technical perspective and management perspective.
   Management wise, we were able to learn how to separate a general goal into multiple sub-tasks and distribute time between them accordingly. This was crucial as the coding part took a long time and went through multiple adjustments during the process. We could not have spent so much time on this if the first part involving laying and deriving the theoretical foundation was not completed on a fairly tight schedule.
   Technical Wise, we had to learn knowledge in new fields such as Simulink and AI-GYM in order to satisfy the requirements of our goals. Simulink was crucial in building our simulation model while the custom AI-GYM environment was used to recreate our MDPs, and describe each state depending on the agent's possible actions.

B. Documentation

1) Code Location:
All original code has been left in a private repository which the mentor and team members all have access to.
2) Data and Documentation:
Detailed documentation including meeting reports, datasets and the manuals are left in a google drive shared folder.

REFERENCES

[1] Messina Z, Olarewaju O. Pressure Controlled Ventilation. [Updated 2021 Aug 2]. In: StatPearls [Internet]. Treasure Island (FL): StatPearls Publishing; 2022 Jan-. Available from: https://www.ncbi.nlm.nih.gov/books/NBK555897/

[2] Ashworth L, Norisue Y, Koster M, Anderson J, Takada J, Ebisu H. Clinical management of pressure control ventilation: An algorithmic method of patient ventilatory management to address ”forgotten but important variables”. J Crit Care. 2018 Feb;43:169-182. doi: 10.1016/j.jcrc.2017.08.046. Epub 2017 Sep 6. PMID: 28918201.

[3] Mohammad Jaber, Lara Hamawy and Mohamad Hajj-Hassan et al. MATLAB/Simulink Mathematical Model for Lung and Ventilator. DOI: 10.1109/ICM50269.2020.9331820

[4] Al Naggar, N.Q. (2015). Modelling and Simulation of Pressure Controlled Mechanical Ventilation System. Journal of Biomedical Science and Engineering, 08, 707-716.

[5] Yan Shi, Shuai Ren, Maolin Cai, Weiqing Xu, Qiyou Deng, “Pressure Dynamic Characteristics of Pressure Controlled Ventilation System of a Lung Simulator”, Computational and Mathematical Methods in Medicine, vol. 2014, Article ID 761712, 10 pages, 2014. https://doi.org/10.1155/2014/761712

[6] H. Y. Al-Hetari, Y. Alginahi, M. N. Kabir, N. Q. Al-Naggar, M. A. Al-Rumaima and M. M. Hasan, “Modeling Lung Functionality in Volume-Controlled Ventilation for Critical Care Patients,” 2020 IEEE 2nd International Conference on Artificial Intelligence in Engineering and Technology (IICAIET), 2020, pp. 1-4, doi: 10.1109/IICAIET49801.2020.9257851.

[7] Nicholas Renotte. 2021. OpenAI-Reinforcement-Learning-with-Custom-Environment. [accessed 2022 Mar 25]; https://github.com/nicknochnack/OpenAI-Reinforcement-Learning-with-Custom-Environment/blob/main/OpenAI%20Custom%20Environment%20Reinforcement%20Learning.ipynb

[8] Jordi TORRES.AI. 2020. Deep Q-Network (DQN)-II Experience Replay and Target Networks.[accessed 2022 Mar 25]; https://towardsdatascience.com/deep-q-network-dqn-ii-b6bf911b6b2c

[9] Adam Paszke. 2022.Reinforcement Learning(DQN) Tutorial.[accessed 2022 Mar 25]; https://pytorch.org/tutorials/intermediate/reinforcement_q_learning.html

[10] Phil Tabor. 2021. DeepQLearning [accessed 2022 Apr 7]; https://github.com/philtabor/Youtube-Code-Repository/blob/master/ReinforcementLearning/DeepQLearning/simple_dqn_torch_2020.py

[11] Shawhin Talebi. 2020. The Wavelet Transform An Introduction and Example [accessed 2022 Feb 28]; https://towardsdatascience.com/the-wavelet-transform-e9cfa85d7b34

[12] Mora Carpio AL, Mora JI. Ventilator Management. [Updated 2021 May 7]. In: StatPearls [Internet]. Treasure Island (FL): StatPearls Publishing; 2022 Jan-

[13] J. Dellamonica et al., ”PEEP-induced changes in lung volume in acute respiratory distress syndrome. Two methods to estimate alveolar recruitment,” Intensive Care Medicine, vol. 37, no. 10, pp. 1595-1604, 2011.

[14] Silva PL, Rocco PRM. The basics of respiratory mechanics: ventilator-derived parameters. Ann Transl Med. 2018 Oct;6(19):376. doi: 10.21037/atm.2018.06.06. PMID: 30460250; PMCID: PMC6212352.

[15] Gattinoni L, Collino F, Maiolo G, Rapetti F, Romitti F, Tonetti T, Vasques F, Quintel M. Positive end-expiratory pressure: how to set it at the individual level. Ann Transl Med. 2017 Jul;5(14):288. doi: 10.21037/atm.2017.06.64. PMID: 28828363; PMCID: PMC5537121.

[16] Johns Hokokins University School of Medicine. 2022. Vital Signs (Body Temperature, Pulse Rate, Respiration Rate, Blood Pressure) [accessed Feb 7th] https://www.hopkinsmedicine.org/health/conditions-and-diseases/vital-signs-body-temperature-pulse-rate-respiration-rate-blood-pressure

[17] Sembroski E, Sanghavi D, Bhardwaj A. Inverse Ratio Ventilation. [Updated 2021 Aug 2]. In: StatPearls [Internet]. Treasure Island (FL): StatPearls Publishing; 2022 Jan-. Available from: https://www.ncbi.nlm.nih.gov/books/NBK535395/

[18] Mark A. Warner, Bela Patel. 2013. Mechanical Ventilation https://www.sciencedirect.com/topics/medicine-and-dentistry/peak-inspiratory-flow

[19] Khoo, M.C.K. (2001) Physiological Control Systems: Analysis, Simulation, and Estimation. IEEE Press Series on Biomedical Engineering, New York, 1-319.