Automatic Mechanical Ventilation Control (AMVC)

Haoyu Shi (hshi25@jhu.edu)
Jiahe Xu (jxu109@jhu.edu)
Mentor: Dr Antwan D. Clark (aclark66@jhu.edu)

Project Review

Good news: Things are on track!!!

1. Building a mathematical model which produces data simulating the PCV signals and the lungs' responses.(Completed)
2. Devise Markov Decision Processes(MDPs) to model these states and the possible actions.(Completed)
3. Building our MDPs in an AI-GYM environment.(Completed)
4. Currently, we need a well-designed cost function for the RL environment(working on).
5. We aim to develop an Reinforcement Learning agent that is able to adjust input parameters for a Pressure Controlled Ventilator in order to help patients’ lungs achieve some certain states(what to do next).

Goals

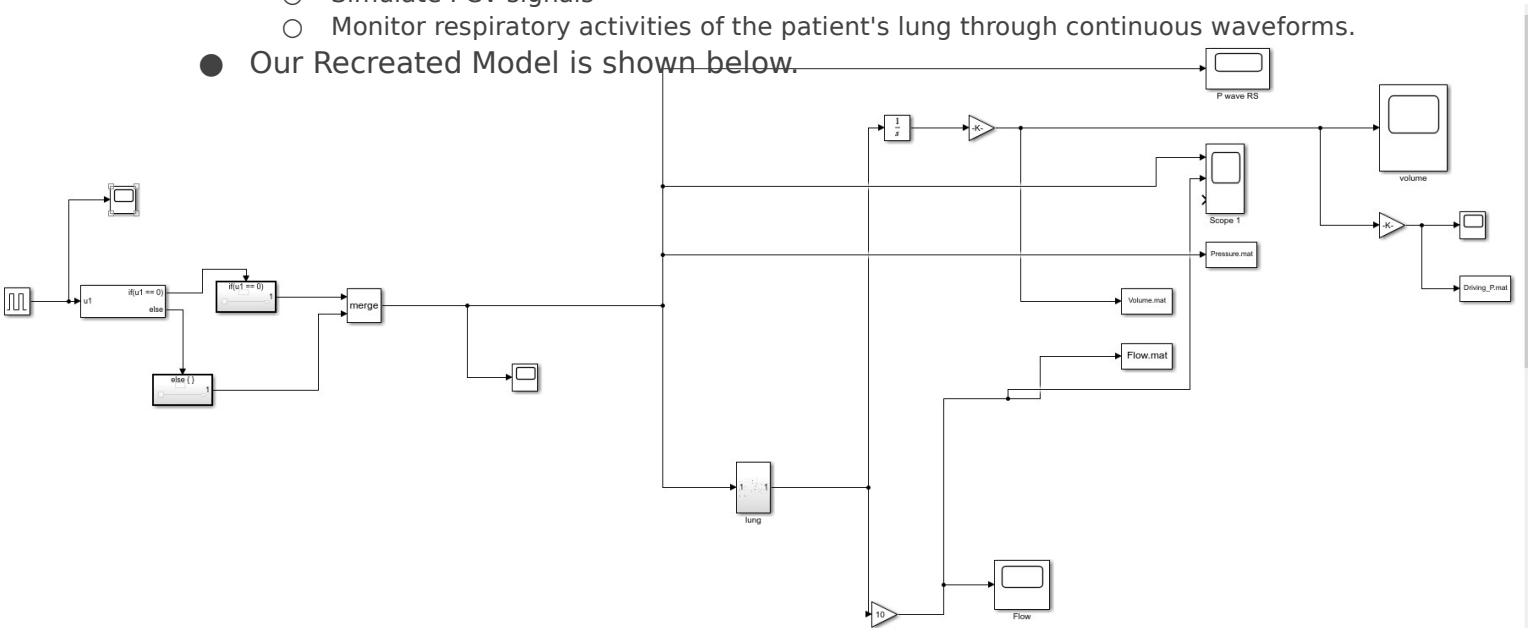
1. Initial Goals are currently still the same.
2. Build an AI-GYM environment to represent different states for the patient.
3. Explore and train an RL agent(ventilator controller)
4. Apply a reliable and suitable evaluation metric for the results.

Dependency Updates

Name	Method	Alternative	Impact	Status
MIMIC III Dataset	Apply for access	Simulation	Affect Training Results	Approved
Computer for Training Datasets	Google Cloud	NA	Unable to finish training	What to do next
Interface between simulink and python	Self Constructed	Rebuild lung model in python	Need a week or two to migrate the code.	Completed and Tested

Reconstruction of Lung Model

- As a preparation step for building the Markov Decision Process(MDP) and deciding the rewards, we will be using a model from Al Nagger's Paper in 2015.
- The model has the following features:
  - Built in Simulink on Matlab
  - Simulate PCV signals
  - Monitor respiratory activities of the patient's lung through continuous waveforms.
- Our Recreated Model is shown below.

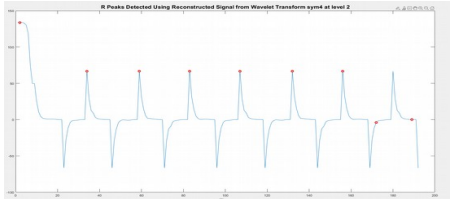


Simulation Result and Waveform Analysis

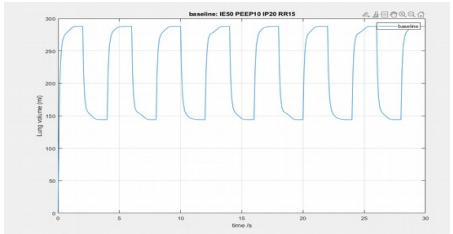
From the Model, we were able to gain data for the Flow, Volume and Pressure. These were collected and graphed.

We applied wavelet transforms and collected the distinct characteristics such as the range of each condition.

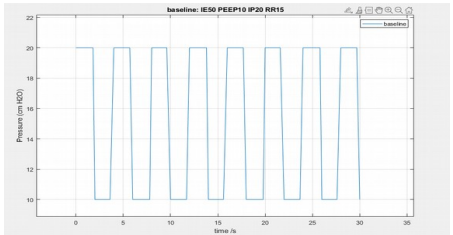
- Figure 1 shows the Max and Min points detected through a wavelet transformation method called Rpeak from an article by Talebi, S.
- Figure 2 is the graph of one set of Volume Data,
- Figure 3 is the graph of a set of Pressure Data.
- All three graphs were monitored during the 30 second period after the setting is applied.



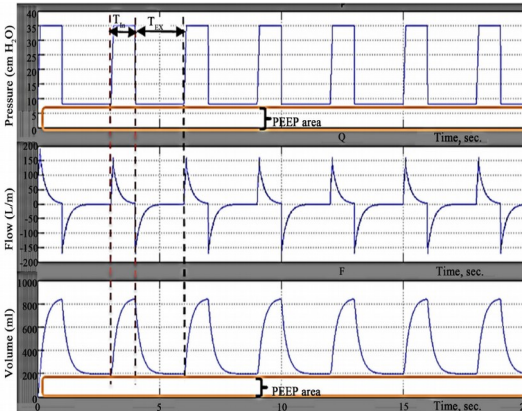
1



2



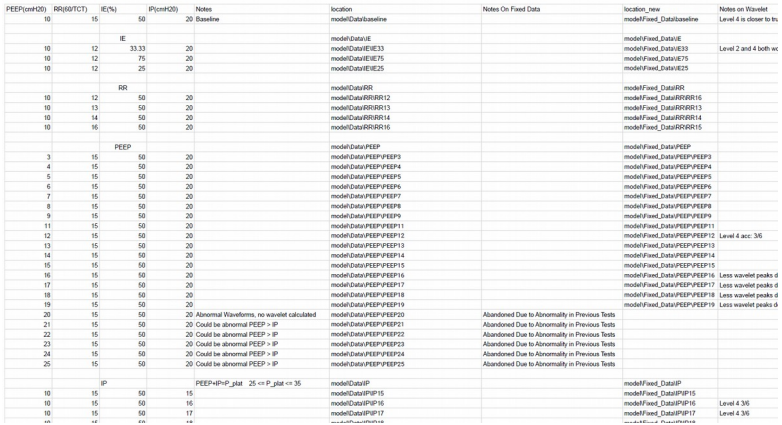
3



Result from [1]

Collected data

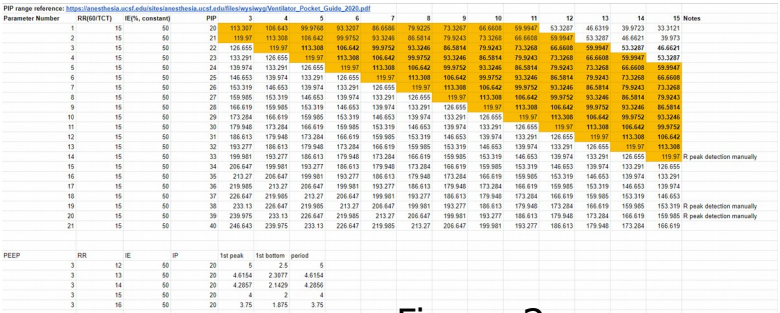
Figure 1



PEEP(cmH20)	RR(60/TCT)	IE(%)	IP(cmH20)	Notes	location	Notes On Fixed Data	location_new	Notes on Wavelet
10	15	50	20	Baseline	model\Data\baseline		model\Fixed_Data\baseline	Level 4 is closer to tru
		IE			model\Data\IE		model\Fixed_Data\IE	
10	12	33.33	20		model\Data\IE\IE33		model\Fixed_Data\IE33	Level 2 and 4 both wo
10	12	75	20		model\Data\IE\IE75		model\Fixed_Data\IE75	
10	12	25	20		model\Data\IE\IE25		model\Fixed_Data\IE25	
		RR			model\Data\RR		model\Fixed_Data\RR	
10	12	50	20		model\Data\RR\RR12		model\Fixed_Data\RR\RR16	
10	13	50	20		model\Data\RR\RR13		model\Fixed_Data\RR\RR13	
10	14	50	20		model\Data\RR\RR14		model\Fixed_Data\RR\RR14	
10	16	50	20		model\Data\RR\RR16		model\Fixed_Data\RR\RR15	
		PEEP			model\Data\PEEP		model\Fixed_Data\PEEP	
3	15	50	20		model\Data\PEEP\PEEP3		model\Fixed_Data\PEEP\PEEP3	
4	15	50	20		model\Data\PEEP\PEEP4		model\Fixed_Data\PEEP\PEEP4	
5	15	50	20		model\Data\PEEP\PEEP5		model\Fixed_Data\PEEP\PEEP5	
6	15	50	20		model\Data\PEEP\PEEP6		model\Fixed_Data\PEEP\PEEP6	
7	15	50	20		model\Data\PEEP\PEEP7		model\Fixed_Data\PEEP\PEEP7	
8	15	50	20		model\Data\PEEP\PEEP8		model\Fixed_Data\PEEP\PEEP8	
9	15	50	20		model\Data\PEEP\PEEP9		model\Fixed_Data\PEEP\PEEP9	
11	15	50	20		model\Data\PEEP\PEEP11		model\Fixed_Data\PEEP\PEEP11	
12	15	50	20		model\Data\PEEP\PEEP12		model\Fixed_Data\PEEP\PEEP12	Level 4 acc: 3/6
13	15	50	20		model\Data\PEEP\PEEP13		model\Fixed_Data\PEEP\PEEP13	
14	15	50	20		model\Data\PEEP\PEEP14		model\Fixed_Data\PEEP\PEEP14	
15	15	50	20		model\Data\PEEP\PEEP15		model\Fixed_Data\PEEP\PEEP15	
16	15	50	20		model\Data\PEEP\PEEP16		model\Fixed_Data\PEEP\PEEP16	Less wavelet peaks d
17	15	50	20		model\Data\PEEP\PEEP17		model\Fixed_Data\PEEP\PEEP17	Less wavelet peaks d
18	15	50	20		model\Data\PEEP\PEEP18		model\Fixed_Data\PEEP\PEEP18	Less wavelet peaks d
19	15	50	20		model\Data\PEEP\PEEP19		model\Fixed_Data\PEEP\PEEP19	Less wavelet peaks d
20	15	50	20	Abnormal Waveforms, no wavelet calculated	model\Data\PEEP\PEEP20	Abandoned Due to Abnormality in Previous Tests		
21	15	50	20	Could be abnormal PEEP > IP	model\Data\PEEP\PEEP21	Abandoned Due to Abnormality in Previous Tests		
22	15	50	20	Could be abnormal PEEP > IP	model\Data\PEEP\PEEP22	Abandoned Due to Abnormality in Previous Tests		
23	15	50	20	Could be abnormal PEEP > IP	model\Data\PEEP\PEEP23	Abandoned Due to Abnormality in Previous Tests		
24	15	50	20	Could be abnormal PEEP > IP	model\Data\PEEP\PEEP24	Abandoned Due to Abnormality in Previous Tests		
25	15	50	20	Could be abnormal PEEP > IP	model\Data\PEEP\PEEP25	Abandoned Due to Abnormality in Previous Tests		
		IP		PEEP+IP<P_plat 25 <= P_plat <= 35	model\Data\IP		model\Fixed_Data\IP	
10	15	50	15		model\Data\IP\IP15		model\Fixed_Data\IP\IP15	
10	15	50	16		model\Data\IP\IP16		model\Fixed_Data\IP\IP16	Level 4 3/6
10	15	50	17		model\Data\IP\IP17		model\Fixed_Data\IP\IP17	Level 4 3/6

Displayed to the right is part of the data we have collected during the process. They are all documented in multiple spreadsheets and in their original files(.mat files).

- Figure 1 shows a documentation of the parameters, distinct features and notes.
- Figure 2 shows the process of calculating and selecting the boundaries and the legitimate values for each output.



PIP range reference: https://anesthesia.ucsf.edu/sites/anesthesia.ucsf.edu/files/wysiwyg/Ventilator_Pocket_Guide_2020.pdf

Parameter Number	RR(60/TCT)	IE(%, constant)	PIP	3	4	5	6	7	8	9	10	11	12	13	14	15	Notes
1	15	50	20	113.307	106.643	99.9768	93.3257	86.6586	79.9225	73.3267	66.6608	59.9947	53.3287	46.6319	39.9723	33.3121	
2	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	
3	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	
4	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	
5	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	
6	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	
7	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	
8	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	
9	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	
10	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	
11	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	
12	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	
13	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	
14	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	R peak detection manually
15	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	
16	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	
17	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	
18	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	
19	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	R peak detection manually
20	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	R peak detection manually
21	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	

PEEP	RR	IE	IP	1st peak	1st bottom	period
3	12	50	20	5	2.5	5
3	13	50	20	4.6154	2.3077	4.6154
3	14	50	20	4.2857	2.1429	4.2856
3	15	50	20	4	2	4
3	16	50	20	3.75	1.875	3.75

Figure 2

Constructing Markov Decision Process

In order to build a custom AI-GYM environment, the MDPs need to be devised first.

We have built two MDPs and they are displayed below:

- Figure 1 is one of the MDPs which models the states and actions for Flow and Pressure.
  - They are merged because the limiting boundaries and included information are either the same or included.
  - Red States are warning States where the input parameters could cause an abnormal Flow/Pressure Value
- Figure 2 are MDPs that model the states and actions for Volume.
  - Each MDP means a value which affect a certain aspect of the Volume value.

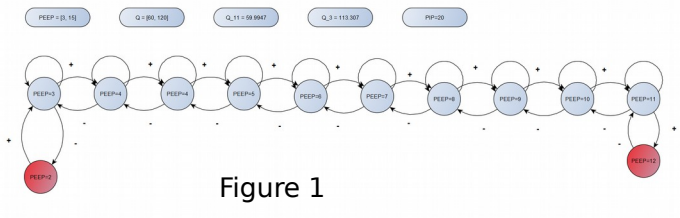
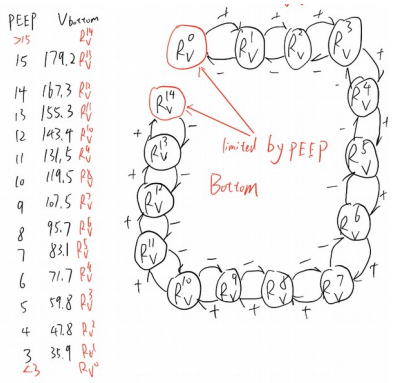
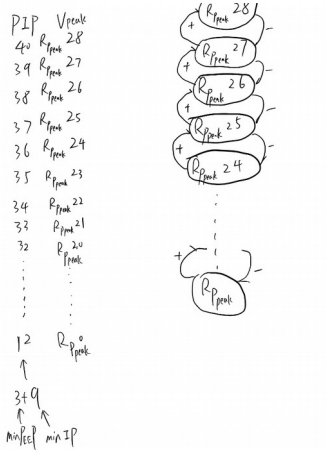


Figure 1

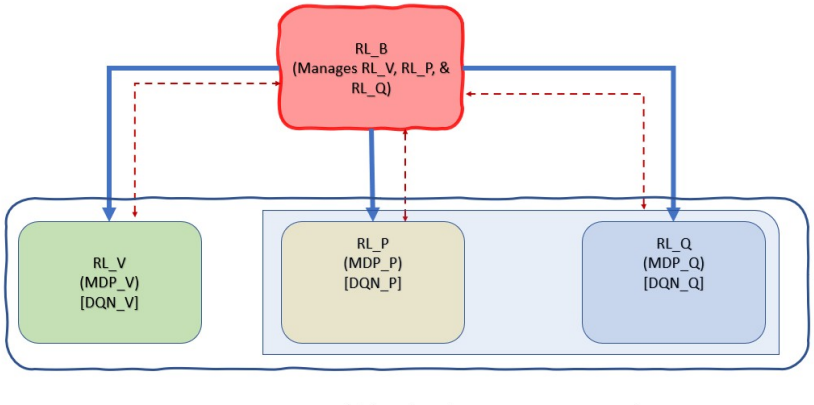




Structure of the agents

We are planning to use a hierarchical design for the agents.

- The individual agents will control the states of the Flow, Volume and Pressure.
- The managing agent would be an agent which is based on a fixed set of rules.



Hierarchal (Deep) Reinforcement Learning Approach

Construction of AI-GYM Environment

We constructed the MDPs listed in the previous slides.

It will be divided into three slides, the actions, the states and a part of the actual implementation

Action Space

The action space of the MDP for Flow Rate is an integer from 0 to 5. The meaning of each value is listed below.

Action	Ventilator Parameter Adjustment
0	PEEP - 1
1	PEEP
2	PEEP + 1
3	PIP - 1
4	PIP
5	PIP + 1

Construction of AI-GYM Environment

Observation Space:

- This will be the legit values for all states.
- Each state is a numpy array containing three elements representing PEEP, PIP and IP(inspiratory pressure).
  - IP can also be calculated by PIP - PEEP
  - IP is an important indicator which determines whether the flow is in safe boundaries.
- The range for each value is listed below

Variable Name	Low	High
PEEP	3	15
PIP	20	40
IP	9	18

Implementation

Figure 1 shows Part of the implementation which contains all elements discussed in the previous slides.

Figure 2 shows samples of the action space and observation space.

```
class MDP_Q_Env(Env):
    def __init__(self):
        # Actions we can take, PEEP/IP down, Remain, PEEP/IP up
        self.action_space = Discrete(6)
        # Peak and Bottom array
        self.observation_space = Box(low=np.array([3, 20, 9], dtype=np.int32), high=np.array([15, 40, 18], dtype=np.int32))
        # self.observation_space = Discrete(12, start=np.array([3, 20], dtype=np.int64))
        # Set Baseline
        self.state = (3, 20, 17)
        # Set Ventilation Length(Optional)
        self.vent_length = 60

    def step(self, action):
        # Apply action

        PEEP, PIP, IP = self.state
        if action > 2:
            PIP += action - 3 - 1
            IP += action - 3 - 1
        else:
            PEEP += action - 1
            IP += action - 3 - 1
        self.state = (PEEP, PIP, IP)
        #Optional Time length
        self.vent_length -= 1
        reward = 0
        IP = PIP - PEEP
        baseline = {'PEEP':10, 'PIP':20, 'IP':10}
        # Calculate reward
        if PEEP >= 3 and PEEP <=15:
            current_reward = 20- abs( PEEP - baseline['PEEP'] )
            reward += current_reward
        else:
            reward += -60

        if PIP >= 20 and PIP <= 40:
            current_reward = 20- abs( PIP - baseline['PIP'] )
            reward += current_reward

        else:
            reward += -60

        if IP <=18 and IP >= 9:
            reward += 20
```

Figure 1

```
[ ]  env = MDP_Q_Env()
     print(env.observation_space.sample())
     print(env.action_space.sample())
```

```
[14.573119 35.311165 10.273634]
3
```

Figure 2

Exploration of Reinforcement Learning Methods

Figure 1

```
model = Sequential()
model.add(Flatten(input_shape=(1,) + states))
model.add(Dense(16))
model.add(Activation('relu'))
model.add(Dense(actions))
model.add(Activation('linear'))
print(model.summary())
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                 Output Shape              Param #
=================================================================
 flatten_1 (Flatten)          (None, 3)                 0

 dense_2 (Dense)              (None, 16)                64

 activation_2 (Activation)    (None, 16)                0

 dense_3 (Dense)              (None, 6)                 102

 activation_3 (Activation)    (None, 6)                 0

=================================================================
Total params: 166
Trainable params: 166
Non-trainable params: 0
_________________________________________________________________
None
```

Idea: For each agent, we're employing deep Q-networks in the following fashion:

1. Target Network – This is used for stabilized training.
2. Experience Network – This is used to predict each agent's actions.

Figure 1 is the implemented networks' structure

Figure 2 is the initial results we have achieved.

Note: These results are preliminary, where the goal is to ensure that the implementation is sound. The following steps are to not only enhance this method, but also to develop evaluation strategies.

Figure 2

```
4380/5000: episode: 73, duration: 0.411s, episode steps:  60, steps per second: 146, episode reward: 1014.000, mean reward: 16.900 [-27.000, 55.000], mean action: 1.233 [1.000, 5.000],  loss: 1759.895630, mae: 136.466690, mean_q:
4440/5000: episode: 74, duration: 0.411s, episode steps:  60, steps per second: 146, episode reward: -1669.000, mean reward: -27.817 [-100.000, 55.000], mean action: 1.217 [0.000, 5.000],  loss: 1780.062866, mae: 135.829254, mean_
4500/5000: episode: 75, duration: 0.427s, episode steps:  60, steps per second: 140, episode reward: 3232.000, mean reward: 53.867 [53.000, 55.000], mean action: 1.133 [0.000, 4.000],  loss: 1758.961670, mae: 138.002274, mean_q:
4560/5000: episode: 76, duration: 0.403s, episode steps:  60, steps per second: 149, episode reward: 616.000, mean reward: 10.267 [-27.000, 55.000], mean action: 1.233 [1.000, 5.000],  loss: 1796.244019, mae: 138.721939, mean_q:
4620/5000: episode: 77, duration: 0.431s, episode steps:  60, steps per second: 139, episode reward: -226.000, mean reward: -3.767 [-27.000, 53.000], mean action: 1.217 [0.000, 5.000],  loss: 1683.570068, mae: 138.038010, mean_q:
4680/5000: episode: 78, duration: 0.425s, episode steps:  60, steps per second: 141, episode reward: -1531.000, mean reward: -25.517 [-100.000, 53.000], mean action: 1.217 [0.000, 4.000],  loss: 1690.273315, mae: 135.788147, mean_
4740/5000: episode: 79, duration: 0.418s, episode steps:  60, steps per second: 144, episode reward: -1737.000, mean reward: -28.950 [-105.000, 56.000], mean action: 1.283 [0.000, 4.000],  loss: 1691.822998, mae: 134.981125, mean_
4800/5000: episode: 80, duration: 0.422s, episode steps:  60, steps per second: 142, episode reward: -379.000, mean reward: -6.317 [-102.000, 60.000], mean action: 1.233 [0.000, 4.000],  loss: 1639.587646, mae: 138.529251, mean_q:
4860/5000: episode: 81, duration: 0.411s, episode steps:  60, steps per second: 146, episode reward: 292.000, mean reward:  4.867 [-104.000, 57.000], mean action: 1.567 [1.000, 5.000],  loss: 1649.453979, mae: 142.360657, mean_q:
4920/5000: episode: 82, duration: 0.435s, episode steps:  60, steps per second: 138, episode reward: -4556.000, mean reward: -75.933 [-116.000, 53.000], mean action: 2.383 [0.000, 5.000],  loss: 1656.407349, mae: 139.246384, mean_
4980/5000: episode: 83, duration: 0.441s, episode steps:  60, steps per second: 136, episode reward: -2254.000, mean reward: -37.567 [-180.000, 60.000], mean action: 1.550 [0.000, 5.000],  loss: 1743.357788, mae: 140.894333, mean_
done, took 40.703 seconds
```

Difficulties and Possible Solutions

1. Difficulties:
    a. Developing the boss agent (i.e., RL_B)
        i. In order to put all three MDPs together for the agent, we would like to have another that could control them.
        ii. This requires a value that spans all three outputs Flow, Volume and Pressure.
    b. Rendering Process and Evaluation Metrics are difficult.
    c. We have currently planned for three RL agents in total, which means that we will need to develop interfaces or methods to link up and coordinate the two sub-agents and the managing agent.
2. Solutions:
    a. There are a list of values that we are currently comparing and analyzing including dynamic compliance, Mechanical Energy etc.
    b. We could combine the level of risk to the patient along with the accuracy of the agent actions.
    c. We are aiming to using the managing agent to adjust certain parameters according a fixed set of rules and updating the goals with the sub-agents.

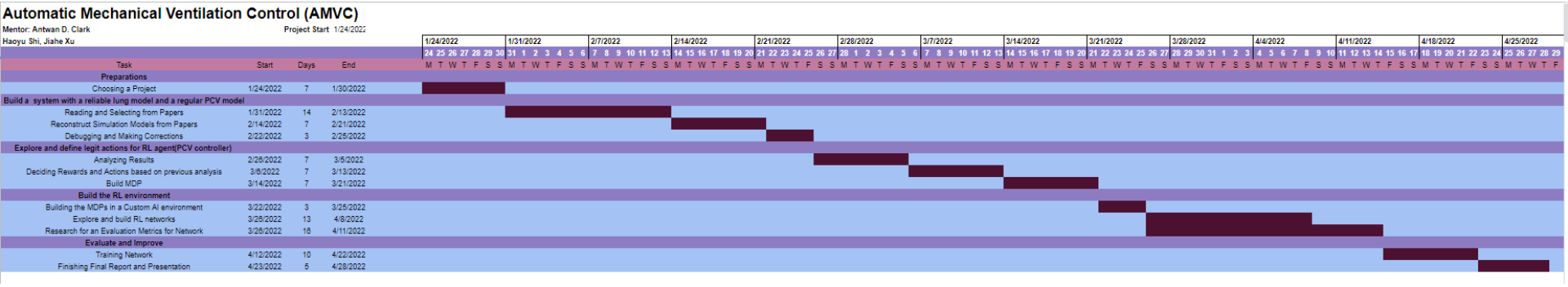
Milestone Progress

Name	Date	Status
Regular PCV model with Lung	2/18/2022	Completed
Explore and define legit actions for RL agent(PCV controller)	3/14/2022	Completed
Build and Evaluate MDP	3/21/2022	Completed
Build the RL environment	3/25/2022	Completed
Exploration of RL networks and methods	4/6/2022	In Progress
Train a RL agent and find metrics to evaluate its performance	4/14/2022	In Progress
Evaluate and Improve Model	4/28/2022	Not Started

Updated Timeline

The timeline has been mostly the same and no long delays have occurred up till now. The only adjustment made was due to an early completion of environment and thus giving time for an important part that is currently in progress. We would also like to put the exploration of evaluation metrics at the same time period of the network search as a metric is crucial for us during the selection of networks.

A more detailed version can be found through the link: https://docs.google.com/spreadsheets/d/1cUyruluDUhf72e8jPAVjTiejxp6aKMbwmo368sQOXZw/edit?usp=sharing



Documentation and Management

1. Code and Notes will be kept in a github repository
2. Reports, Graphs and Specific Data generated are uploaded to a shared folder in google drive where the mentor and all team members can access.
    a. Each step and its results are specified in a spreadsheet.
    b. Every meeting information, notes and reports are all included as well.
3. Meetings are still maintained at twice a week on Mondays and Fridays with Dr Antwan D. Clark.

References

1. Al Naggar, N.Q. (2015). Modelling and Simulation of Pressure Controlled Mechanical Ventilation System. Journal of Biomedical Science and Engineering, 08, 707-716.
2. Mohammad Jaber, Lara Hamawy and Mohamad Hajj-Hassan et al. MATLAB/Simulink Mathematical Model for Lung and Ventilator. DOI: 10.1109/ICM50269.2020.9331820
3. Choudhary, A. (2020, April 27). Deep Q-learning: An introduction to deep reinforcement learning. Analytics Vidhya. Retrieved March 27, 2022, from https://www.analyticsvidhya.com/blog/2019/04/introduction-deep-q-learning-python/

Thank you!