Question 5.

I'd like to pick FiremanBot, which is kind of clearer in functions and definition. Below, I'd give 5 situation where FiremanBot might need different abilities. After each situation, I'd explain some algorithms that might be of use

Situation 1.

- Consider our FiremanBot works on a flat terrain with blocks that FiremanBot can't pass through. Also, our FiremanBot knows where the the target is. It needs to find a shortest path to save the target.

- Because we only have one FirmanBot, so we can not do BFS. DFS is not efficient in finding a shortest path. So a better solution is to use A-star Algorithm. Because Fireman knows where the target is and where it is. We can generate heuristic using either Euclidean distance or Manhattan distance in order to implement A-star algorithm.

Situation 2.

- Consider one traveler is trapped on the peak of a mountain. We need to send our FiremanBot to find this guy. (Our FiremanBot does not know the exactly where it is and where the traveler is. It only knows the traveler is on the peak of the mountain.

- With the same limitation: we only have one FiremanBot, so we can not do Beam search. However, the hill-climbing algorithm is easily trapped in a local maximum (the peak of some convex terrain). So we are supposed to use simulated-annealing algo. With the probility of picking a lower neighbor, FiremanBot has the ability to avoid local minimum to some extend.

Situation 3.

- Consider there is a one-floor house catching fire, and humans can not get in. We need to send FiremanBot in to detect if there is still someone inside. The FiremanBot need to keep away from fire in order to stay functional.

- Because the FiremanBot does not have the prior knowlege of where is catching fire, So it needs to have the ability to make inference from the current situation and its database. Like what we discussed in the lecture, we can give our brave FiremanBot the initial KB: if [i,j] is catching fire, then the four blocks [i-1,j], [i+1,j], [i,j+1], [i,j+1] gets high temperature. Each time FiremanBot detects high temperature, it queries its KB, finds one way to move on, and update its KB.

Situation 4.

- the same as situation 3, now we have a camera on FiremanBot that takes grayscale photos. And we want to get color photos in order to get better understanding of the situation.

- This situation is the same as our project: colorization. It is related to neural networks. First we should build a neural network (maybe CNN, is a good choice). Then based on the dataset of photos taken in place catching fire, we train our model and continually modify the weight of neurons (which are filters in CNN). Finally we feed the network with the photos taken by FiremanBot and get the colorized photos

situation 5.

- Consider the following situation. We have a place full of 3 sizes of rooms: large / medium / small room. Target is in one of these rooms and is not known. The FiremanBot is able to search every room. However, the bigger the house is, the hard the target can be found. This situation is like what we meet in project 3. And we can set a similar false negate rates:

$$P(\text{Target not found in room}_i \mid \text{Target in room}_i) = \begin{cases} 0.1 & \text{small room} \\ 0.3 & \text{medium room} \\ 0.5 & \text{large room} \end{cases}$$

- This situation is related to Hidden Markov Model. The point is to ~~generate~~ model the knowledge probalistically. And then use this knowledge to ~~make~~ generate future action. This specific problem is a Filter: $F_{t+1}(x) = \beta \Sigma F_t(x') P(x|x')$. If we add observation in as extra info, we can also fit in observation model to the Filter formula.

Finally, thanks to our brave Fireman Bot!

Bonus As for Bonus, is your dog Luna Rose dressed in a Superman Costume? I'm not sure. And unfortunately, I'm not a good painter...