Assignment 1

Student Name- Garima Garima

Student ID - 200315571

Question 1: Your task is to design and deploy a search-and-rescue agent that controls a group of robots to search for survivors in a large collapsed building after the earthquake. The robots communicate with each other using wireless mobile communication nodes they carry. They use the same equipment to communicate with a central server to report the necessary findings. Communication quality depends on the terrain and other environmental factors. The robots must search for potential survivors in the environment and report survivors' location to human rescuers.

1. **Define a PEAS specification for the agent. Justify your choices.**
   Ans: Search and Rescue(SAR) is an intelligent agent that controls robots who are searching for survivors in the building.
   Assumptions:
   1. SAR agent is aware of the floor plan or structure of the building
   2. SAR agent is keeping track or the explored and unexplored areas of the building
   3. Robots have some medical knowledge to identify if human is alive or not
   4. Robots have cameras and GPS for object detection and navigating in the building
   5. Robots will communicate correct position of potential survivor to SAR agent
   6. Robots are able to do complex navigation through disaster material
   7. Sensors cannot see through the walls and so does cameras

Performance Measure	We can measure the performance of search and rescue operation by considering by below written factors: ▪ Number of rescued live victims ▪ Correct location of survivor ▪ Optimal use of all the search robots ▪ Time taken to scan through the complete building
Environment	In this case environment will be the collapsed building and everything that would be inside the building at that time. Below are specifications of the environment: ▪ This will be *partially observable* environment as sensors in real life cannot scan through the wall or obstacle, so robot cannot see what could be behind of the obstacle. ▪ Our intelligent agent should be able to keep track of other robots through a wireless communication device. This environment will *be non-deterministic* environment, as intelligent system is controlling the robots, and if there is an obstacle on the way and robot cannot see through the obstacle and is not able to move the obstacle out of the way, agent would not be able to determine the next action for robot. ▪ This is a Multiagent environment, as there are many robots and agent involved and ones actions would affect another robot. ▪ This environment is a *sequential* environment as each decision of agent and next state depends upon current state and the input provided by the robots. Agent should be able to plan the next steps for robot, as if one robot has already checked in particular location another robot does not have to go there again to scan. ▪ This environment can be considered *static* if we assume that nothing is moving and there is not probability that a ceiling could fall, or wall could fall, but the same environment can be *dynamic* also if there is some water leakage in building after earthquake, or moving something from way could lead to another disaster in the building. ▪ This environment can be *continuous*-state and continuous-time problem, the state of robot depends on if they have detected human or any obstacle and state of other robots, continuous change of location of different robots. ▪ This environment will be *known* as agent is already provided with structure or floor plan of the building so it can intelligently see which robot is in which area, in which conditions we can say human is alive or dead

Actuators	**Search Robots –** Search robots are following the instructions provided by SAR agent, so they are actually actuators for the agent. Search robots on other hand have many actuators like *motors, wheels* to move from one direction to another, *activation of medical sensor* on detection of human to check if human is alive or dead, *sending a communication* about potential survivor to agent using wireless mobile communication device.
Sensors	**Sensing Robots –** Search robots are acting as sensor of agent, as they are sensing objects and sending information about them to agent. Robots on other hand have many sensing devices such as *cameras* for *object recognition* that object is a human or anything else, they have *medical diagnosis sensor* also to sense activities of human like heart rate, blood pressure or any other important aspect

2. **What is the minimum agent necessary for this task? Justify your answer with respect to the task.**
   Ans: Search and Rescue agent task majorly deals with managing the robots, optimizing the use of them, for this purpose model based agent can also satisfy the requirement. They keep the internal state of the area that they cannot see now. In this case agent can keep internal state of the area that has been explored by the robots and keep on updating it. Updating the internal state requires two kinds of knowledge to be encoded in the agent program. First, we need some information about how the world evolves independently of the agent – for example, if one robot has scanned area in north from its standing position, it would have scanned area in different direction after some time. Second, we need some information about how the agent's own actions affect the world – for example, that if agent asked robot to cover some area that means now there will not be any survivor there and there is no point of sending some other robot again there. Model based agent works well in partial observable environment, they can take good guesses of the location that is not observed now, as suppose robot checked in one room moved in another, even though old room is not observable now but model based agent would know that there is no survivor in that area. Simple reflex agent would not work here since we need to keep track of area and need to remember different activities different robots are doing. Model based reflex agent would work well in this case.

3. **How would the environment and the design change if the building is prone to fire? Justify your answer**
   Ans: If the building is prone to fire, our environment would change drastically and we would have to change the design of our agent and environment to accommodate effect of fire in the process. Below would be changes in the environment:
   1. Environment would become dynamic, as there could be fire at any time, some areas could be on fire and some of the areas could still be out of the reach of fire. There may be somethings that could increase the fire, change the direction of fire. Our agent would be unaware about those scenarios.
   2. Environment now becomes unknown environment, as even though agent knows about the floor plan and medical sensors but density of fire, direction of fire and other things agent would not be able to predict and agent's state of knowledge becomes unknown in this scenario. Agent would need to learn how fire may behave in order to plan the next action.

   In order to accommodate fire into the design we could make below changes:
   1. If we have robots that are fire proof and water proof and can navigate inside to find the survivor, or know the limit that that up to what extent human can survive in fire like this.
   2. Robot can carry some amount of water in it and agent should be able to take a call that this amount of fire can be controlled by throwing water and should ask robot to do so.
   3. SAR Agent should be able to figure out the direction of fire and ask robots to move in that direction first to check for potential survivor and move them from the way.
   4. Robots should be able to detect level of fire, if it is high they should communicate back to the central server.
   5. SAR Agent should be given some prior knowledge about the fire and thigs that can aggravate them or reduce them so it can communicate the same to robots.

4. **If the agent chosen was a learning agent what would the critic function evaluate?**
   Ans: Critic function: If the agent is a learning-based agent we would need a critic function which would evaluate the performance of our agent. The critic function would evaluate the performance parameter which needs to be optimized for example the efficient use of each robot and others in our case. All the robots are working together performing their own tasks on commands of SAR agent to maximize the number of rescued live victims. Critic function could give information written below to maximize performance:
   1. Area covered till now. This should be maximum.
   2. Time been taken to save the survivor after its detection. This should be minimum.
   3. Number of times two robots went to same place for checking, which should not happen.
   4. Time being spent human detection that was not alive. This should be minimum.
   5. Wrong object detection. That is robot perceived something as human when it was not human.
   6. How many times location of survivor was not being send correctly. This should be minimum.
   7. Time elapsed should be minimum.

   If learning agent is been fed with above kind of information, it can learn from its mistakes and perceive them and try to maximize and minimize things accordingly.

5. **State one weakness of the original design and how would you fix it.**

   Ans: One of the weakness in the original design is that if a robot detects the potential survivor they would have to send information back to the human rescuers which could take time if communication is not going super-fast. We never know what the situations can be by the time human rescuers comes. So, if a robot detects the human and that could be potential survivor, robot should have a capability to pick up the human, give that information back to agent and agent should guide the robot with best possible path to bring the human in safe place. They should themselves be able to pick and drop human when injury levels are high and human cannot wait for human rescuers to come. In scenarios like this even if there is an obstacle SAR agent can ask robot to move the obstacle in some other place that wouldn't harm anything and then robot can move further, if robot cannot do that then agent won't be able to tell robot on next actions.

Question 2: Suppose you are running the Amazon products recommend system. This recommend system mines data from 1) Purchased shopping cart data 2) Items added to cart but abandoned 3) Pageviews. All of the data feed into the recommendation engine to help predict what your customers are most likely to buy next. Over time as you 'feed' the engine more data, it gets smarter and smarter with its recommendations so that your email subscribers and customers are more likely to engage, click and buy. Under this circumstance, answer each of the following questions with a paragraph or more of text.

1. **Define a PEAS specification for the agent.**

   Ans:

Performance Measurement	We can measure the performance of recommendation system but below considering the below factor: ▪ Increase in number of sale coming from customers purchasing recommended product ▪ Accuracy , means number of correct recommendations out of total recommendations ▪ Number of users added recommended item to Wishlist. This could mean that product had relevance and user may buy it later. ▪ Number of email subscriber who clicked on the recommended link sent in the email
Environment	In this case environment will be our retail website, location of the user, time of the year(festivals, seasons), purchase history of the user, age of the user, gender of the user, time being spent in particular category of products and many other factors. Below are specifications of the environment. ▪ This environment is *partial observable*. Even though recommendation agent have the information about users purchase history, time being spent on product, what people are buying at that location, what things people purchase together and can recommend product on basis of above information. Still we cannot say what exactly user is looking for at that moment and without that recommendation agent cannot really recommend the relevant product which make is partial observable. ▪ This will be a *multi agent* environment as users action will affect the action to be taken by agent. Agent will have to see what product user is currently looking at and then do relevant recommendations. ▪ This environment is *sequential*, as system will see what product person is looking for and what are relevant items to that product. In episodic agent generally does not care about long term effects of the action but that is not the case here. ▪ This environment is *stochastic* as next state depends on what is the current state (that is the product user is looking) and action by agent (related product recommendations). Next state could be a product from the recommendation window, but we are not certain about that as user may or may not click on the product. So, the next state is not completely certain. ▪ This environment will be *dynamic* as the state can be changed by agent or by user. As state will be changed by agent only when user click on the product recommended by the agent not otherwise. So, state created by agent and next possible state to be created by agent can be interrupted when user clicks on something else rather than state(product) agent suggested. ▪ This is a *discrete* environment, as agent will recommend only discrete number of products and will change when user is looking for different product

	▪ This is a *known* environment if agent is given with the data what to recommend or agent knows how to recommend. If agent is not provided with any prior information about any product then this becomes *unknown* environment.
Actuators	▪ Sending an email **:** Sending email to email subscriber for recommendation of products on some basis would be actuator. ▪ Showing the recommended product when user is purchasing the product or viewing the product in recommendation section.
Sensors	There are many sensors for recommendation system: ▪ What product or product category user is currently searching for ▪ Which product user is spending most time on, reading reviews and photos ▪ Where does user clicks on the screen

2. What is the minimum agent necessary for this task? Justify your answer.

Ans: A model based agent is minimum agent required for this task. As agent needs to have a percept history of what user have already purchased and what product user is purchasing now and many other parameters, after considering all of that then system should recommend a product to the user. System should also keep an internal state of what product recommendation user have marked not interested or do not recommend in future and not to show those items to user. System should keep on updating the internal state also. This model agent can give good guesses looking at current and internal state when system encounters some unknown state. If we use a simple reflex agent in this case, then it will just look at the current state that is what exact product user is looking at on the screen and give recommendation of that product as mapped in the table. But this is not the case when it comes to recommendation system, system looks at major three things purchase history, things in cart and abandoned, pageviews, simple reflex agent cannot do that. So, we need at least model based agent.

3. How would the environment and the strategy of your agent change if there are no initial data for some new users?

Ans: If there is no initial data for user there will be some changes in environment and in agent. Environment becomes *unknown* for the agent. As agent would not know what to recommend and it may have to try different options to create a percept history for those users. Since there is no data available for user, system can start recommending on the basis of location, gender, deals, most purchased, unique items, latest items in the stock. After that user may start interacting with system and agent can start creating the percept history for the user.

4. If the agent chosen was a learning agent what would the critic function evaluate?

Ans: Critic functions are used by learning agents which provides constant feedback to the agent so agent can improve its performance function. In our case critic function could give information about following:

1. What new products are being purchased with what items
2. What product users asked for not to recommend
3. What product user are never buying together
4. Which products user actually bought when showed in recommendations

These values could be shown to the agent so agent can improve and never show or start showing those products in recommendation.

5. State one weakness of the original design and how you would improve upon it.

Ans: One weakness of the design according to me is that if system is just looking at the previous purchased items, items abandoned in cart and pageviews, it may not be able to recommend best of the products. Just looking at previously purchased items, agent can recommend the things which were purchased quite some time ago and maybe now it's time to repurchase them example medicines or deal on abandoned items in cart or if it was according to pageviews, user know those items exists and may not want to purchase them. To improve this model we need to feed data about many different things, like neighbourhood data means, in some proximity what people are purchasing, seasonal items, items on flash sale, big day sale coming, items with huge and best number of reviews and different things. When we put this kind of data also in system and in recommendation then I can say that the system is actually giving good recommendations to user so user will be more engaged in the webpage and is most likely to purchase the item.

Question 3: For each of the following agents specify the features of the environment and justify each feature choice with a 1-2 sentence answer:

1. Video recommendation:

- **Fully vs Partial Observable**: This is *partial* observable environment because agent does not always know what video or type of genres user is looking for and what to recommend.
- **Single vs Multi Agent:** This is a *multiagent* environment, as recommendation does not depend only on agent but it depends upon what genres or what type of videos user generally watch. So, two parties are involved.

- **Deterministic vs Stochastic:** This is *stochastic* environment as next state depend upon on current state and action taken by agent. We can say next state could be video from recommended videos but we cannot be certain that user will watch the recommended video. We can increase the certainty of next state but still is not completely certain.
- **Episodic vs Sequential:** This is *sequential* system as action of agent that is recommending a new video depends upon what video currently user is watching or most of the users are watching at that time and next state could depend upon the video recommended by our agent and further more. Action of agent can affect the future states which makes it sequential.
- **Static vs Dynamic:** This is *dynamic* environment, because the next state does not only depend upon action taken by agent but on user also. In static environment nothing changes between state and when agent is deciding what actions to take but here it can change.
- **Discrete vs Continuous:** This is *discrete* environment as there are only finite number of actions that can be done. Agent have to give the recommendation and then it has to wait for user's next action.
- **Known vs Unknown:** This is *known* environment to agent. As agent knows the current state, it knows about all the videos and it know what user is currently watching or looking for and recommend the same.

2. **Qudra-drone autopilot:**
   - **Fully vs Partial Observable:** This will be a *partial observable* environment as there could be tall building or trees which may block the way.
   - **Single vs Multi Agent:** There could be birds, clouds, helicopters or other things that can fly and agent would have to take actions according to them which make it a *multi agent* environment.
   - **Deterministic vs Stochastic:** Next state depends upon current state and agents action and we can to some extent know about next state that could be a deterministic environment, but if there is a sudden battery failure or any hardware failure, an obstacle comes in state makes it a *stochastic* environment.
   - **Episodic vs Sequential:** This is an *sequential* environment. Actions that drone is taking would affect its future actions too. If agent have a goal to go somewhere and it start going in different direction then it would affect its further decision to fulfil the goal.
   - **Static vs Dynamic:** This is a *dynamic* environment. As when agent is taking an action there could be change in state for example of some kind of object comes into picture which will change the decision of the agent action.
   - **Discrete vs Continuous:** This is a *continuous* environment as agent have to take continuous actions and decide in which direction to go and what to do. Agents decision cannot be distributed in discrete time space.
   - **Known vs Unknown:** This is a *known* environment as agent is aware of the schedule, current time, tracks to follow, and other factors.

3. **Virtual customer support:**
   - **Fully vs Partial Observable:** This is *fully observable* environment, as agent is completely aware of users query and where it has to go and look for answer.
   - **Single vs Multi Agent:** This is *multiagent* environment as action of agent depends upon what questions are been asked by user and it have to act accordingly. In single agent environment action of agent does not depend or get effected by anything.
   - **Deterministic vs Stochastic:** This is *stochastic* environment as we are not certain what next state is going to be. Agent can support user by answering the query but after that next state depends upon user, that user wants to continue asking questions or want to end the sessions. So, next state is still not certain after knowing current state and action by the agent.
   - **Episodic vs Sequential:** This could be *sequential* environment as the action by Virtual Customer agent depends upon what question is been asked by the user and on the basis of action by agent user will take next action that would be next state which makes it sequential as future states are affected by agent's actions.
   - **Static vs Dynamic:** This is a *dynamic* environment as when agent is looking for suitable answer of a query asked by user, agent still needs to keep listening or looking that if user has changed the query (state changed)or asked new query so agent can find the relevant answer.
   - **Discrete vs Continuous:** This is *discrete* environment as agent once performs its action have to wait for user input and then take new action.
   - **Known vs Unknown:** Environment will be *known* to agent, as agent knows where to look for the query user asked, as we provide agent with information makes it as known environment to agent.

4. **Subway automatic dispatch system:**
   - **Fully vs Partial Observable:** This is a *fully observable* environment, as subway knows the track and it knows at what time it have to leave the station. It should know the schedule of at what time where should subway be. All communications of accidents will be communicated back to the dispatch agent so it can plan for the corresponding actions.
   - **Single vs Multi Agent:** This is a *single agent* environment as there is only one agent single dispatch system that is handling different subways in the environment.

- **Deterministic vs Stochastic:** This is a *stochastic* environment as next state will depend upon current state and action taken by the agent and in this case we may know current state that it is the time for subway to close the door and leave the metro station, but because of some technical failures it did not start, in that case it becomes stochastic.
- **Episodic vs Sequential:** This is a *sequential* environment because if one subway is broken down or not on time it would affect the other subway running on same platform.
- **Static vs Dynamic:** This is *dynamic* environment as there could be a case that subway was about to be dispatched but then suddenly there was some technical issues and delayed its dispatch from station and now the state has changed and agents action needs to be changed.
- **Discrete vs Continuous:** This is a *discrete* environment as dispatch system will only have to choose from limited number of actions that either to dispatch the subway or not to dispatch on the basis of time.
- **Known vs Unknown:** Environment is *known* to agent, as it is completely aware of time, schedule of subways, which subway would be at what station platform.

5. **Siri response system:**
   - **Fully vs Partial Observable:** This will be a *fully observable* environment as this agent will act on the basis of query been asked by user or another system. Agent is fully aware of what is been asked and where to look for the answer.
   - **Single vs Multi Agent:** This is *multiagent environment* because state depends not only action of Siri but also depends upon what is been asked by another agent, human or system.
   - **Deterministic vs Stochastic:** As this is a multiagent environment and this conversation is derived by user but not our agent and agent will have to act according to queries. This makes the system *stochastic*. We can always tell the next state on based of question asked and action of agent but can never be sure.
   - **Episodic vs Sequential:** This is a *sequential* system. Since agent have to act on the basis of query being asked by user and user may further ask questions dependent on answer provided by agent. Action of user does affect the next state which makes it sequential.
   - **Static vs Dynamic:** This is *dynamic* environment, as agent is communicating with external system or user and conversation keeps on evolving, state is getting changed on the basis of agent reply and user question rather only by agent. If state is changed only by agent's actions then we can say it is static environment but not in this case.
   - **Discrete vs Continuous:** This is a *discrete* environment as once agent provide answer, it has to wait for user response if any or end the conversation.
   - **Known vs Unknown:** This is a *known* environment, although agent does not what user is going to ask but it does know the rules that on the basis of query by user agent have to find appropriate answer. It is not like agent does not know what user is expecting and it starts giving random answers. Agent know what is been asked, it just have to find the best possible answer.