People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research

Amar Telidji University – Laghouat

FACULTY: Technology

DEPARTMENT: Electronic

MASTER THESIS

: Presented by

Nadir Djelmani

FIELD: science and technology

FILIERE: Electronic

SPECIALTY: Electronics of embedded systems

Theme

UAV adaptive control based on AI to transfer the energy wirelessly.

Juries of thesis:

First and Last name	Grade	Quality
Mr. Noureddine Chaib	M.C.A	President
Mr. Amuer Abdelkader Ilyes	M.A.A	Examiner
Mr. Omar Sami Oubbati	M.C.A	Advisor

Promotion :2021

THANKS

Thanks to Almighty ALLAH for giving me strength and the ability to complete this thesis work.

I would also like to take the opportunity to thank my supervisor DR. OMAR SAMI OUBBATI for his help and proper guidance throughout my research.

Thanks to all the teachers who supported me in my studying path, thanks to all the colleagues and friends, Thanks to all my family who has believed in me and stood by my side.

Thank you.

Abstract:

"This thesis aims to configure a set-up for UAV self-controlling without human intervention and no prior information about the environment to recharge IoT devices wirelessly, so they continue to operate for as long as possible. IoT devices have become more popular and widespread, and their involvement is crucial in many fields. Also, they can provide a multitude of services to enhance human lives. As challenges, UAV is supposed to explore the environment to get more information about it continuously, and therefore we should carefully minimize its energy consumption to avoid its possible failure. As a solution, an AI-based algorithm called Q-learning is adopted to optimize the movement of the UAV and enhance the energy transfer toward the IoT devices.

Key Words: UAV, Q-learning, IoT."

Résume :

"Cette thèse vise à configurer une installation pour l'autocontrôle des drones sans intervention humaine et sans aucune information préalable sur l'environnement pour recharger les appareils IoT sans fil, afin qu'ils continuent à fonctionner aussi longtemps que possible. Les appareils IoT sont devenus de plus en plus populaires et répandus, et leur implication est cruciale dans de nombreux domaines. En outre, ils peuvent fournir une multitude de services pour améliorer les vies humaines. Comme défis, l'UAV est censé explorer l'environnement pour obtenir plus d'informations à son sujet en continu, et nous devons donc soigneusement minimiser sa consommation d'énergie pour éviter son éventuelle panne. En guise de solution, un algorithme basé sur l'IA appelé Q-learning est adopté pour optimiser le mouvement de l'UAV et améliorer le transfert d'énergie vers les appareils IoT..

Mots clés : UAV, Q-learning, IoT"

ملخص:

"تهدف هذه الأطروحة إلى تكوين إعداد للتحكم الذاتي للطائرات بدون طيار دون تدخل بشري ولا توجد معلومات مسبقة عن البيئة لإعادة شحن أجهزة إنترنت الأشياء لاسلكيًا ، بحيث تستمر في العمل لأطول فترة ممكنة. أصبحت أجهزة إنترنت الأشياء أكثر شيوعًا وانتشارًا ، وأصبح مشاركتها أمرًا بالغ الأهمية في العديد من المجالات. أيضًا ، يمكنهم تقديم العديد من الخدمات لتحسين حياة الإنسان. باعتبارها تحديات ، من المفترض أن تستكشف الطائرات بدون طيار البيئة للحصول على مزيد من المعلومات عنها بشكل مستمر ، وبالتالي يجب علينا تقليل استهلاكها للطاقة بعناية لتجنب فشلها المحتمل. كحل ، تم اعتماد خوارزمية قائمة على الذكاء الاصطناعي تسمى Q-Learning لتحسين حركة الطائرات بدون طيار وتعزيز نقل الطاقة نحو أجهزة إنترنت الأشياء.

الكلمات الرئيسية: UAV, Q-learning, IoT"

List of Contents

List of Figures:

List of Tables:

List of Abbreviations:

2D Two Dimensions.

A

AI Artificial intelligent.

ARE Adaptive and Random Exploration.

B

BS Base Station.

F

FAA Federal Aviation Administration

H

HLP High Altitude Platform.

HALE High Altitude Long Endurance.

I

IoT Internet of Things.

L

LAP Low Altitude Platform.

LRRT Local Rapidly Random Tree

M

MALE Medium Altitude Long Endurance.

Q

QoE Quality of Experience.

R

RF Radio Frequency.

RL Reinforcement Learning.

RPAS Remotely Piloted Aircraft Systems

rUAV receiver Unmanned Aerial Vehicle.

T

tUAV transmitter Unmanned Aerial Vehicle.

U

UAV Unmanned Aerial Vehicle.

UAS Unmanned Aerial system.

W

WPT Wireless Power Transfer.

General Introduction.

20.4 billion is the number of IoT devices in 2020, according to Gartner [1]. It is a vast number, and it keeps increasing daily. Those IoT devices include simple and complex objects, such as smartphones, smartwatches, industrial equipment, and connected vehicles. Generally, IoT devices exchange information with the cloud and between them, depending on the service they are executing. Like any paradigm, IoT devices also face issues that we need to address for better performance. One of the significant problems is the limited capacity of IoT devices in terms of energy. Indeed, they have a limited battery power that will not last for too long, especially with the new generations of mobile networks that require more energy. To fix this problem, a new technology called Wireless Power Transfer (WPT) could recharge IoT devices wirelessly and ensure IoT devices' proper functioning for an extended period. It is a challenging issue due to the dynamic of IoT devices in the environment; therefore, we propose a flying energy source that transfers the energy to the IoT devices wirelessly.

When it comes to flying objects in technology, we think of Unmanned Aerial Vehicles (UAVs). In the last few years, UAVs have been used in various domains. Due to their popularity and low costs, the UAVs are deployed in civilian services such as delivery service, object tracking, natural phenomena, and more applications. Telecommunications domains also use UAVs, such as in public events like sports games and festivals. For instance, UAVs are used as flying hotspots to improve internet speed or be the primary source of the Internet when there is an electrical issue with the terrestrial internet source, such as in an earthquake [2]. UAVs come in two types according to their performance altitude [3]. High Altitude Platforms (HAP) UAVs, where their altitude can exceed up to 20 km; one of the uses of this type is to provide internet access in large zones, the second type is Low Altitude Platforms (LAP), also named as Drones. Telecommunication domains use LAP UAVs to

maximize the transferred energy or data. Unfortunately, UAVs also have batteries on board with a limited lifetime;this leaves us with two main questions, how to get the UAVs to fly over IoT devices and transfer the energy? Moreover, how to charge the maximum number of IoT devices?

To manipulate the UAV navigation, we use real-time path planning, which is a flexible method, and the UAV has no prior knowledge about the environment. It is mainly used in intelligent situations like these and also used in robotics. It is more attractive due to its efficiency and ability to work in a new unknown environment; this method's dynamic reaction to the environment base on the newly generated information from the environment. We provide a learning ability to the UAV to control its behavior and reactions to the environment. Using Q-learning with a bit of knowledge about the environment (possible actions to take), the UAV learns and improves its behavior.

Three chapters compose this thesis:

- The first chapter presents IoT devices and how they are involved in our daily lives, the IoT applications. Then we present UAVs, their different types, their applications and how UAVs are assisting the IoT environment.
- In the second chapter, we present methods to control UAVs using Q-learning. We study four applications that use Q-learning to control UAVs.
- In the last chapter, we describe our proposed method to control our UAV and compare it to other methods discussed in the previous chapter.
- Finally, we conclude our thesis.

Chapter I IoT network.

I.1 Introduction:

An IoT network is a collection of intelligent devices wirelessly connected among them or connected to the cloud with the Internet. An IoT network provides many services without human involvement; millions of IoT networks surround us daily in different fields such as industry, civilian environments, and hospitals [4]. Because of IoT devices' various uses and vast fields, there is no specific definition of what an IoT device exactly involves [5]. As we mentioned before, IoT devices face issues such as the estimated battery life and data transferring. Here comes the critical role that UAVs play in IoT domains because of the flexible features the UAV can offer to the IoT device, such as energy transmission, data gathering and active internet hotspots [6].

I.2 IoT definition:

Internet of things (IoT) includes all the objects that can access the Internet like sensors, smartwatches, and intelligent vehicles. These IoT devices could be interconnected among themselves or to the cloud using standard protocols [7]. With IoT devices, human lives can be a lot easier and improved; an IoT device replaces many manual commands [8]. For instance, a temperature sensor connected with an air conditioner controls the room's temperature, an alarm connected to the house's lights manages the lightning in the house or a door lock connected to the house's security base. All these improve human life in different ways like security, comfortability, health. The first one who used the term IoT was Kevin Ashton from the United Kingdom in 1999 [9]. He defined the system as multiple devices physically connected to the Internet. IoT has become very popular and widespread that even sensors, smart devices, and smart objects can access the Internet.

Enormous intelligent devices are connected to the Internet and work together from different locations. In addition to the connected devices that are popular such as computers, smartphones, smartwatches. There are also cities, glasses, cars, and sensors connected to the Internet [10]. Various domains deploy IoT devices; lately, most companies' services and electrical sensors are connected devices. These devices interconnect without human intervention, such as in roads smart sensors that gather information and other devices that control the traffic lights. IoT has significant potential in helping aging societies in transportation and mobility, with the help of the Internet, we can access any of these devices from any place [11], they save a lot of time for human and enhance their life, and gather a lot of data and make it available on the Internet for all the IoT devices around the world. By increasing the amount of data, the more data we gather the more precision we get [12]. The IoT is described by a combination of physical (Hardware) and digital (Software) components. Because of the reliable memory, high technology of microprocessors, the power management, IoT performance is well productive and it can digitalize functions in many domains [13].

IoT technology is usually a combination of three core layers [14] (c.f. Figure I-1):

1. The device layer: the IoT specified hardware such as sensors, processors, actuators, and in some cases, additional software to organize the data before sending them to the cloud, etc. This part does the physical functionality.
2. The communication layer: the IoT communication protocol, to communicate between them or with the cloud.
3. The cloud layer: contains software that does the digital functionality such as data analyzing, processing control. Using the data generated by the IoT device layer.

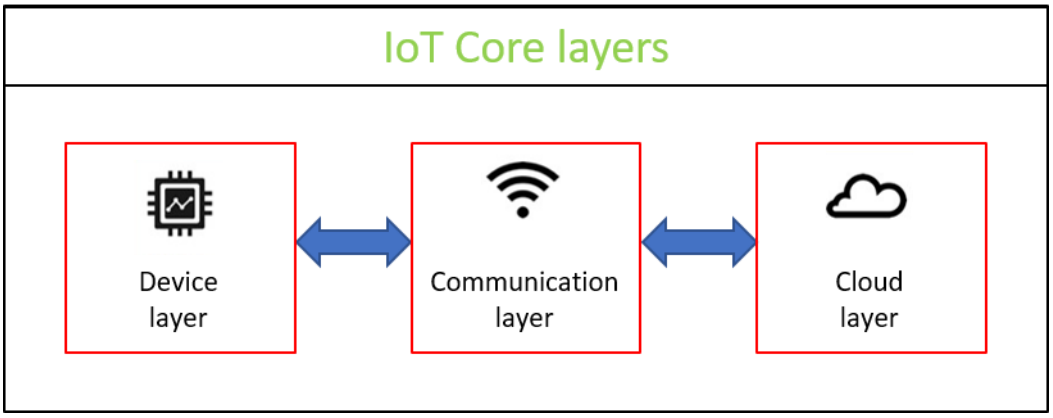


Figure I-1: IoT core layers.

I.3 IoT applications:

The IoT technology is deployed in an enormous number of domains and in different ways for different purposes, because of their benefits and high performance IoT devices are used in many different applications [15], [16].

- **Smart health area:** an enormous number of essential services IoT devices can provide in the healthcare area, such as patient surveilling, and chronic disease management. Some of the advantages in this field are reducing emergency room wait time, remote drug management, alerts, and notifications. All these services and more help realize an improved healthcare management with reduced costs.

- **Smart houses:** IoT devices' services in this domain include temperature controllers, gas sensors, locking systems, surveilling cameras to provide a secured house and lightning control.

- **Smart cities projects:** cities are working toward improving the lives of their population using IoT devices including transportation, real-time monitoring parking areas, the intelligence of lighting streets and pollutions controllers.

I.4 IoT assisted by UAVs:

UAVs offer powerful telecommunications features, such as carrying a base station to enhance wireless access coverage and replace the accidentally broken mobile network antenna. IoT devices also have benefited from UAV features such as recharging the IoT devices wirelessly, gathering data from IoT devices [17].

I.4.1 UAV definition:

UAV was firstly used in military missions, as a type of aircraft. The term UAV refers to an aircraft with no pilot on board it could be controlled from the ground or it can fly autonomous [18]. UAV have been through such a developing path, from the beginning when it was used in military services. It was a large aircraft and costs too much. To the small and cheap version available now, even though every type of UAV has its specific activity and ability, we could use Fixed Wings UAV (FW-UAV) for longer flights and high speed. For flexibility and good stability,we can use Rotating Wings UAV (RW-UAV). Some types of UAVs are called Drones. Drones can be considered as UAVs, but UAVs are not considered as drones [19].

The mostly used common terms are UAV and Unmanned Aerial System (UAS) the different between these two terms is that, UAV stands for the flying platform and UAS refers to boththe flying platform and the ground control station [20]. The popularity of modern UAVs began in 2006 when the first commercial licenses were promulgated by the FAA (Federal Aviation Administration); these licenses have eliminated some rules placed on piloted UAVs. Therefore, the companies started thinking about implementing UAVs in their services. Parrot, a French corporation launched its Parrot (AR UAV) in 2010, which can be handled entirely using a smartphone Wi-Fi. It almost achieved critical results and commercial success immediately. Amazon also launched a delivery system based on UAVs in

2013.Another term is called Remotely Piloted Aircraft Systems RPAS it is a type of aircraft that is controlled remotely by a pilot and this type is excluded from UAV and UAS since the last ones are aircrafts with no pilots on board. The standard UAV components include motors, rotors, speed controllers, batterie, sensors, antenna for connectivity, camera, accelerometer to measure the speed and landing gear (c.f., Figure I-2). Generally, these are all the technological components; there could be more components, or less depending on the service this UAV provides, it can be remotely controlled through its antenna [21].

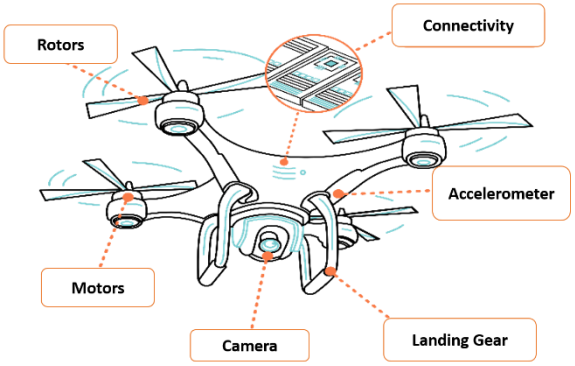


Figure I-2: UAV components.

I.4.2 Types of UAVs:

UAV comes in different sizes and shapes and with different flying methods based on their propellers and size. We characterize UAVs in three categories:

1. **According to the Number of their Propellers:** we set three main types of UAVs sorted according to the number of propellers.

a) **Single rotor UAV:** A single rotor model is consisted of an inside rotor and tail rotor that helps to stabilize the heading (c.f. Figure I-3). It is efficient in high flying, fast flying and in case of hovering while carrying heavy objects single rotor style helicopters could be the best option [22].

Figure I-3: Single rotor UAV.

b) **Multi-rotors UAVs:** It is the most common type of UAV; it is used where stabilization and flexible actions are important such as in filming and objects tracking. They could be categorized according to the number of rotors they is Tricopter UAV which stands for a UAV with three rotors (c.f., Figure I-4.a), Quadcopter UAV with four rotors (c.f., Figure I-4.b), Hexacopter UAVs with six rotors, Octocopter UAVs with eight rotors (c.f., Figure I-4.c). The more rotors, the more energy consumed and vice versa, UAVs with multi-rotors operate in electric motors because of its high precision[23].

a) Tricopter. b) Quadcopter. c) Octocopter.

Figure I-4: Types of Multi-rotor UAVs.

c) **Fixed Wing UAVs:** this type of UAVs has fixed wings and it seems like the old airplanes (c.f., Figure I-5), it cannot stand stable up in the air and it is mostly used for lifting packages [24].

Figure I-5: Fixed wings UAV.

2. **According to their size:** UAVs can be categorized according to their size and each size has its main advantage of use [25].

a) **Small UAV:** it could be used as strong weapon for spying, its size vary from the size of the large insect to about 50cm (c.f. Figure I-6.a).

b) **Medium UAVs:** can carry up to 200kg and have flying time capacity that goes up to 15 Mn (c.f. Figure I-6.b).

c) **Large UAVs:** this type of UAVs is mostly used by military, and its size could be comparable to small airplanes (c.f. Figure I-6.c).

a) Small UAV. b) Medium UAV. c) Large UAV.

Figure I-6: different sizes of UAV.

3. **According to their range:** each type of UAV has its limited altitude of flying [23]:

a) **Close range UAVs:** it weights between 2 and 20 Kg and it can fly up to 1 km, it is used in both civilian and military applications (c.f. Figure I-7.a).

b) **Mid-range UAVs:** it weights between 150 Kg and 600 Kg and it can fly up to 3 km and it is deployed in military applications (c.f. Figure I-7.b).

c) **Long range UAVs:** it is the largest UAV and it weights over 600 kg, in this type of UAV there are two subtypes MALE (Medium Altitude Long Endurance) UAV which can fly up to 14 km (c.f. Figure I-7.c) and HALE (High Altitude Long Endurance) UAV its altitude can exceed up to 20 km (c.f. Figure I-7.d).

a) Close range UAV.

b) Mid-range UAV.

c) MALE.

d) HALE.

Figure I-7: UAVs different range.

I.4.3 Assistance of UAVs in IoT applications:

UAV have an important role in IoT applications, the number of IoT applications keeps increasing daily in various domains such as in military missions, and civilian life improvement and the UAV offer features to help improve these IoT applications.

- **security:** UAV and IoT devices could be used in civilian security applications such as in tracking the criminal or suspect, as a flying alarm system, finding a missing person, or it could be used as a security guard with the help UAVs sensors and with the implementation of AI it can perform really well as a guard [26].

- **Communications**: using a UAV as a hotspot with to provide high signal of connectivity to the ground IoT devices, these devices include smartphones, smartwatches, smart cars, and smart sensors. With high signal and longtime of connectivity these IoT devices will perfectly perform their services [27].

- **Monitoring of tree plantation:** a UAV with smart sensors is deployed to measure the height and crown volume of trees in agriculture field and this data helps controlling the watering system, and generate new information about trees productions, also it is used to monitor planted areas, soil, and field analysis [28].

- **Rockfall trajectory planning:** in earthquake disasters one of the problems is the free-falling rocks from high terrines, in this application a quadcopter UAV carrying IoT devices including high quality camera and sensors have to find the initial rock falling point and then identify the rockfall path [29].

- **Air Quality sensing in smart city**: Air pollution is one of the high environmental health risks, in a smart city UAVs carry sensors and gather information about the air in different locations and transmit the data gathered from these sensors to the ground base station using this data smart cities limit the air pollution and prevent it [30]

I.5 Conclusion:

with the assistance of technology human lives could be easier and safer, the IoT devices are majoring in every field and in different ways in addition to their services IoT devices collect data and share it with the other IoT devices, this helps IoT devices perform better because of the big amount of data. To keep these IoT devices performing for much longer an important newly developed technology is deployed to recharge the IoT devices usingUAVs as power sources, to get these UAVs to recharge the IoT devices the idea of UAV self-controlling should be implemented to control the UAV behavior. Because of the increasingly huge number ofUAVs in different applications, the UAV have to learn by itself and discover the unknown environment using AI techniques.

Chapter II UAV adaptive control system based on AI.

II.1 Introduction:

UAVs can be controlled manually by human using a remote controller, another way is to give the UAV the exact path to go through and all the prior information including directions and distances. Because of the increasingly huge number of UAVs and the developed environment most applications now implement a new method called UAV self-learning path planning, this method is based on AI algorithms and methods to control the UAV's behavior in the environment.

II.2 Q-learning:

Q-learning or Quality-learning, is one of the reinforcement learning's approaches, the general idea of the reinforcement learning is to help the agent interact with the unknown environment and find the optimal decision strategy. Q-learning showed great results in many domains such as video-games and robotics [31]. It is a simple algorithm and its main concept is to give the agent a reward of each action, based on these rewards the agent avoids the actions with less reward. In Q-learning approach there is a set of states S that describe the environment and all the possible states for the agent, a set of actions A the agent act according to this set we give the agent all the possible actions. The agent takes an action $a \in A$, after taking this action the agent receives a reward of this action (c.f., Figure II-1), the q-function uses this reward to calculates the q-values and save them in the q-table by the coordinates (state, action), the size of the q-table is (states * actions) and it is initialized with zeros, these values are updated after each step. This algorithm must maximize the number of rewards at the end [32].

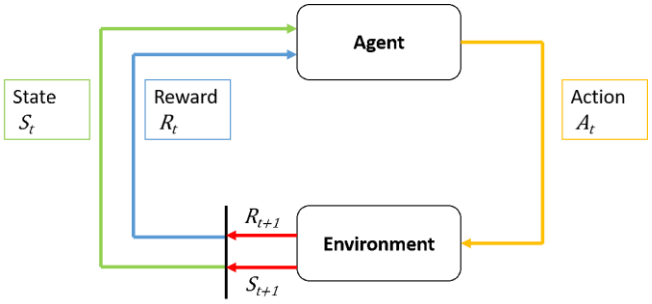


Figure II-1: Q-learning diagram

The components of a reinforcement learning model are (c.f., Figure II-1):

- **Agent:** the object that interact with the environment and receives reward depending on the next state, its main objective is to maximize the number of rewards.
- **Environment:** the area where the agent moves and improve its behavior.
- **State:** the position of the agent and depending on this state the agent gets a reward.
- **Action:** the action that the agent choses to take in every step to move between the states.
- **Reward:** the rewards that the agent gets and its values are different at any application, in order to optimize and improve the agent behavior.

So, the agent according the it is state S_t choses to take the action A_t with a maximum reward, and move to the next state S_{t+1} and get a new reward R_t. The q-values are calculated with the q-function presented below:

$$Q\,(s,a) = (1 - \alpha) * Q\,(s,a) + \alpha\,(R\,(s,a) + \gamma * Q\,(s^{'},a*)) \qquad \textbf{(1)}$$

$Q(s', a*)$ is the optimal q-value of the next state, and α is the learning rate it is a constant that defines how much change is allowed at every q-table update, the high learning rate causes a large change to the q-table and vice versa. γ is the discount factor this constant is used to balance the current reward and future reward, a high discount factor considers the future rewards more heavily [33].

During the process of the agent in learning and behaving there are two types of actions, exploration, and it is when the agent picks a random action from the possible actions and exploitation is when the agent picks an action according to the q-table and the optimal action to pick from the q-table is the action with the maximum q-value. A coefficient named ε or the exploration rate controls which type of action to take, at each step the algorithm generates a random number and compare it to ε if its greater than ε the type of the action is exploitation, if the generated number is less than ε then the type of the action is exploration. After a number of episodes, the agent picks all the action according to the q-table (exploitation).

II.3 Application N°1:

The first application is about a flying energy source and it was published in 2020, in this application a specialized UAVs named rUAVs (receiver UAVs) provide wireless hotspots to the ground nodes. To maximize the period of their performance a UAV named tUAV (transmitter UAV) hovers over the rUAVs and wirelessly transmit the energy to them using WPT technique, so that the rUAVs continue to perform while harvesting the energy from the tUAV. On a 2D geographical area represented as a grid of 3*3 cells, three random cells contain rUAV each rUAV is located at the center of the cell and they remain in their positions during the process. The tUAV is positioned in a random cell at the beginning of each episode of the learning process, it transmits wireless

energy to all the rUAV and the closest rUAV receive the big amount of the energy (c.f. Figure II-2)[34].

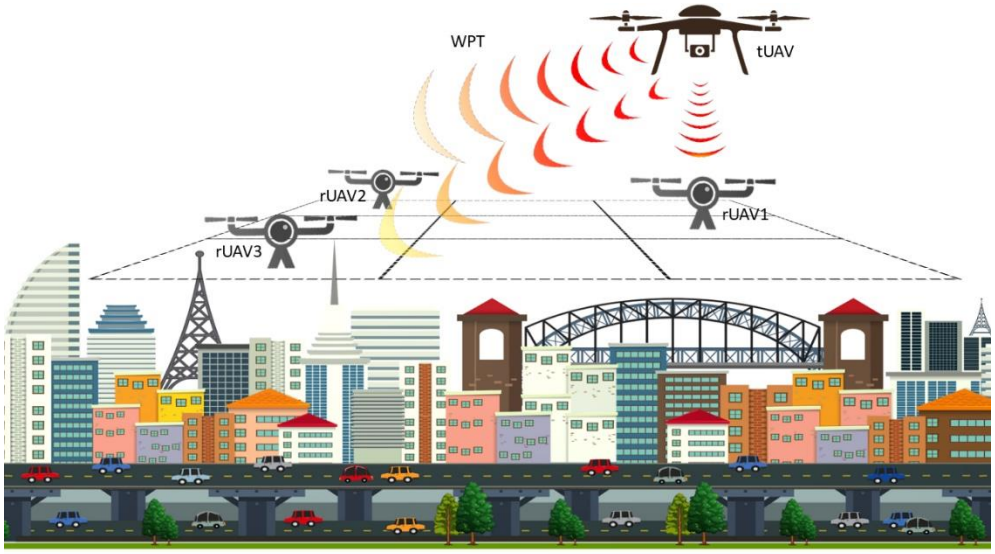
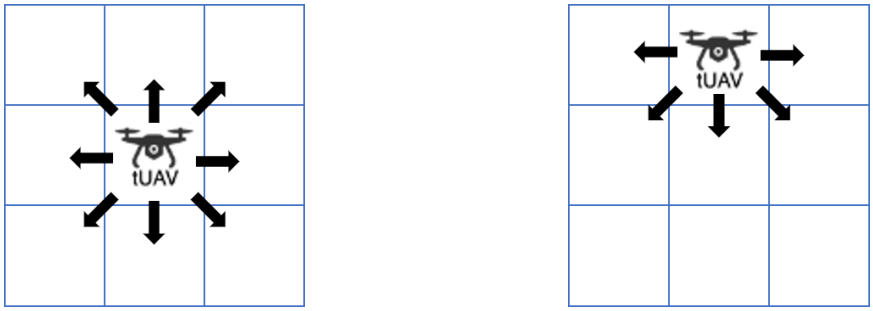


Figure II-2: tUAV recharging rUAV wirelessly.

Some data will be exchanged between the tUAV and rUAVs including the battery level, locations, and the received energy from the tUAV. This energy transfer process is not enough to keep all the rUAVs working for an infinite time. Therefore, when an rUAV reach a low battery level it should leave the environment and recharge its battery wirelessly using WPT technique. Due to the lack of information about the environment the control of the tUAV bases on a Q-learning algorithm to control its behavior and optimize the trajectory in order to maximize the energy transfer and keep the rUAVs performing for as long as possible, the tUAV can move in 9 directions (c.f. Figure II-3.a) but it's not allowed to take actions that might take it out of the grid. Therefore, the number of the available actions changes depending on the state of the tUAV (c.f. Figure II-3.b).

The algorithm runs for 50000 episodes and at the beginning of each episode the rUAV's battery level is initialized with random values between 60 and 100, the tUAV

receives a penalty if one of the rUAV leave the environment to recharge from the base station wirelessly.



a b

Figure II-3: Available actions.

II.3.1 Advantages:

- The simplicity of the approach.
- Maximizing the energy of the UAV by deploying the Q-learning algorithm.

II.3.2 Inconvenient:

- The Q-learning algorithm runs on the tUAV which will cause more energy consumption.
- The environment is not completely unknown there are some exchanged data between tUAV and rUAV such as the location and battery level.
- The energy is transferred widely to all the rUAVs. Only the rUAVs that are closer to the tUAV will receive an efficient amount of the energy, but for the rUAVs that are far from the tUAV most of the transferred energy is lost in the air.
- The number of available actions is variable which make it easy to the tUAV to stay in the environment.

II.4 Application N°2:

The second application is about path planning and obstacles avoidance and it was published in 2017, this application addresses both the UAV navigation and obstacle avoidance using Adaptive and Random Exploration (ARE) approach. While the UAV takes actions and navigates in the environment to reach its final target it must also avoid hitting the obstacles (c.f. Figure II-4), and escape from dangerous situations to a safer path.

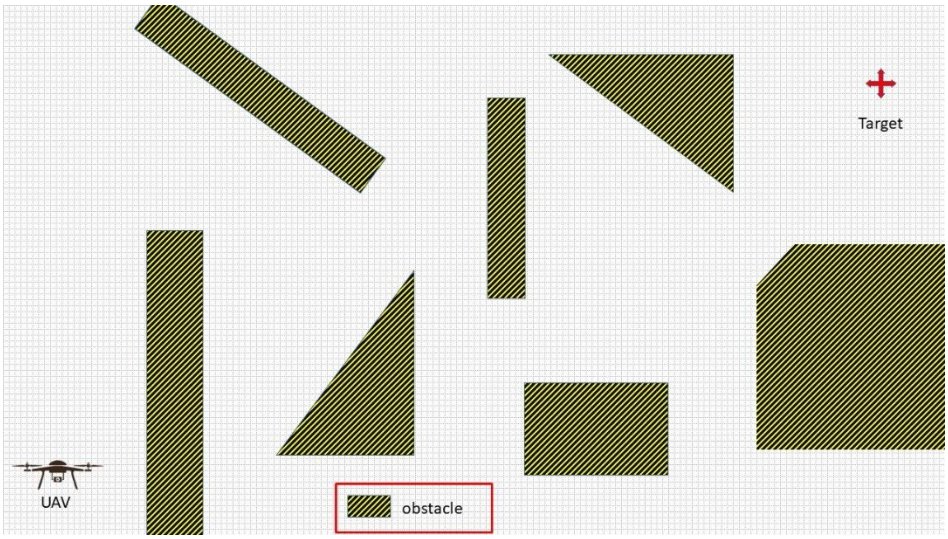


Figure II-4: Obstacle avoidance application environment.

The ARE approach is a composite of learning module, action module, and trap-escape module. The action module takes commands from both the trap-escape module and the learning module and allows the UAV to perform the actions (c.f. Figure II-5). According to the historical data of UAV states and actions the learning module trains the policy of action selection, for each state a learning network calculate the policy of each possible action to take in this state, taking in consider the distance between the UAV and the nearest obstacle and the final target, when the UAV gets closer to the final target then the action selection becomes more important. An arrangement mechanism is

implemented to evaluate the risks and determine if the UAV must escape from the current state, in case if it is near an obstacle, or the UAV should take an action and update the action selection strategy (c.f. Figure II-5), the UAV is implemented with sensors that can detect objects from 0 to 360 degree within 500 cm[35].

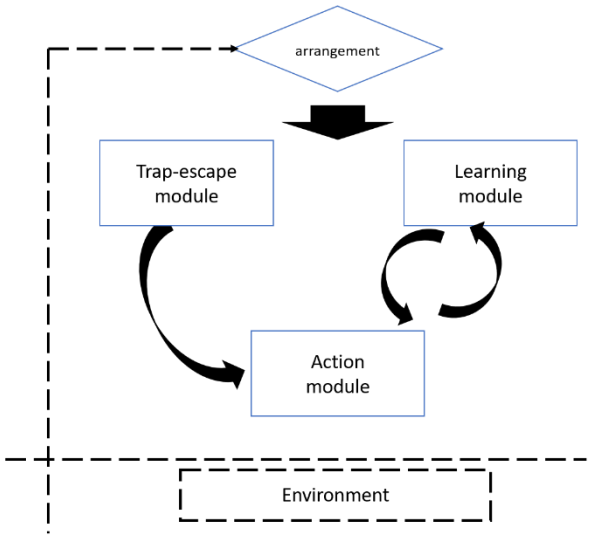


Figure II-5: ARE approach

The functioning process of an ARE approach: we can discretize the ARE approach into 6 steps[35]:

1. Set the UAV's initial state, which includes its location and data repository.
2. Examine the present state in the environment.
3. Determine whether or not the UAV state is in danger. If it is in trouble, use the trap-escape module to get out of it quickly. If not use the action selection module.
4. Chose the optimal action in this state depending on the action selection policies.
5. Take the step and pay attention to the reward and the next state. Using the observed reward and the highest reward of the next state, update the Q-value for the state

6. Set the state to the new one, and redo the process until the UAV reaches the final state.

II.4.1 Advantages:

- Considering the obstacles avoidance is a good point with the daily developed environment.

II.4.2 Inconvenient:

- The process is complicated and uses many algorithms.
- Energy maximization and the recharging process is not taken in consideration.
- Trajectory optimization is not taken in consideration.

II.5 Application N°3:

This application is about providing video streaming service with QoE (Quality of Experience) aware and it was published in 2019. In addition to the Macro-BS (Macro-Base Station) a few UAVs that carry small base stations are deployed to provide video streaming services, the UAVs are used to decrease the connected devices to the Macro-

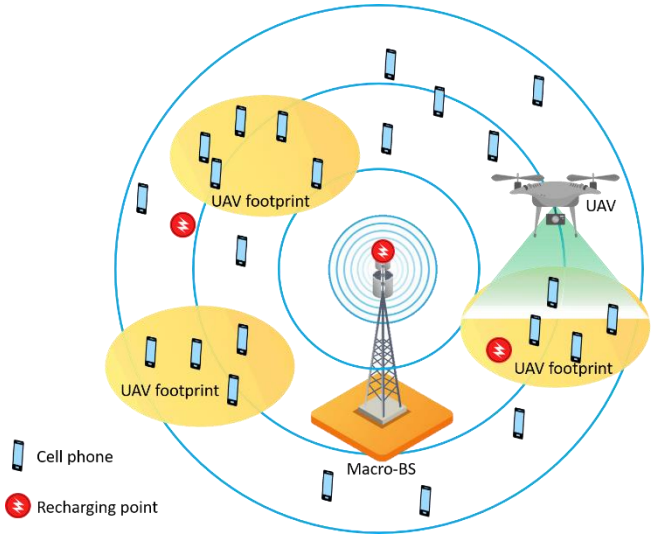


Figure II-6: providing video streaming service.

BS in order to provide good quality videos, video fluidity, and video smoothness for all the users (c.f. Figure II-6). The Macro-BS covers a circle of about 3 km radius. In this area there are users that are represented as parks or meeting points in real life such as in sport games, because of the huge number of users the video streaming service provided by the Macro-BS may not be enough to satisfy all the users, therefore the addition of the UAVs will help improve the service. The UAV can cover an area of 500 m and it has a limited time of flying because of its energy consumption, which is a challenging issue since the UAV control is autonomous.

The behavior of the UAVs is controlled by a Q-square approach, which deploys Q-learning algorithm in addition to the QoE aware, the delay of chunks is set as the quality key metric of the QoE aware. The UAV is free to fly between clusters and recharging points, or remain hovering over a cluster and provide streaming service for predetermined time, the hovering time is limited to increase the UAV exploration and avoid the case where the UAV hovers over one cluster in all the process. The UAV does not serve any user when it is moving, and all the users will connect to the Macro-BS, but when the UAV is hovering over a cluster it provides equal video fluidity to all the users inside the cluster and these users will not be served from the Macro-BS. The UAV must consider recharging its battery therefore after deciding the next action to take the algorithm calculates the residual energy in the next state if the current energy level is not enough to reach the next state, then the UAV must immediately leave to the nearest recharging point[36].

The UAV have to optimize its energy use and find the best cluster to fly to in order to improve the video streaming service for all the users. Using multiple UAVs is considered, when the number of the UAVs on the field are not enough to satisfy the users requests additional UAVs will be added to the field in order to get the best results.

II.5.1 Advantages:

- The importance of the service.
- considering recharging the UAV's battery.
- Multiple UAVs.

II.5.2 Inconvenient:

- Missing the path planning.

II.6 Application N°4:

This application is about using TEXLORE which is a model-based algorithm and it was published in 2017. There are two methods of RL approaches, the first method is the model-free method and it is where the agent uses directly the experience and trial-error approach to calculate the reward and find the optimal actions. The second method is the model-based method and it is where the value function is updated from the environment model the algorithm updates the model according to the newest experience. Most model-

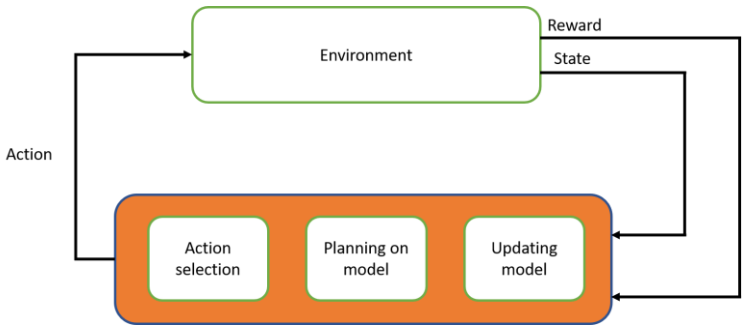


Figure II-7: TEXPLORE diagram

based algorithms are not used in real-time systems due to the intensive calculations that might take some time to finish, TEXPLORE is also a model-based algorithm but to avoid the delay issue, TEXPLORE uses a parallel architecture that as quickly as required takes action based on the current policy without waiting for the model updating process.

This method is composed of model learning and planning modules, and action selection module (c.f. Figure II-7). TEXPLORE learns the environment model using Decision Tree which can help predict the relative change in states transition.

By deploying this algorithm to control a quadcopter UAV on a grid map of 21*41 cells, the UAV must reach the goal using a short path and also manage to find the best location to recharge its battery that are aligned in the top named TRP (Top Recharging Point) and the bottom named BRP (Bottom Recharging Point) of the grid map (c.f. Figure II-8), the UAV has 8 possible actions at the beginning of the episode it start with maximum level of battery and on its path it must at least recharge its battery once and it must reach the final target with the maximum possible level of battery, for an horizontal or vertical action the UAV gets a reward of -1 and for a diagonal action it gets -1.4 as a reward, in case in hits the edges if the grid the UAV receives reward -10 and remain in its position, if the battery drop down to 0 the episode terminates with a reward of -500[37].

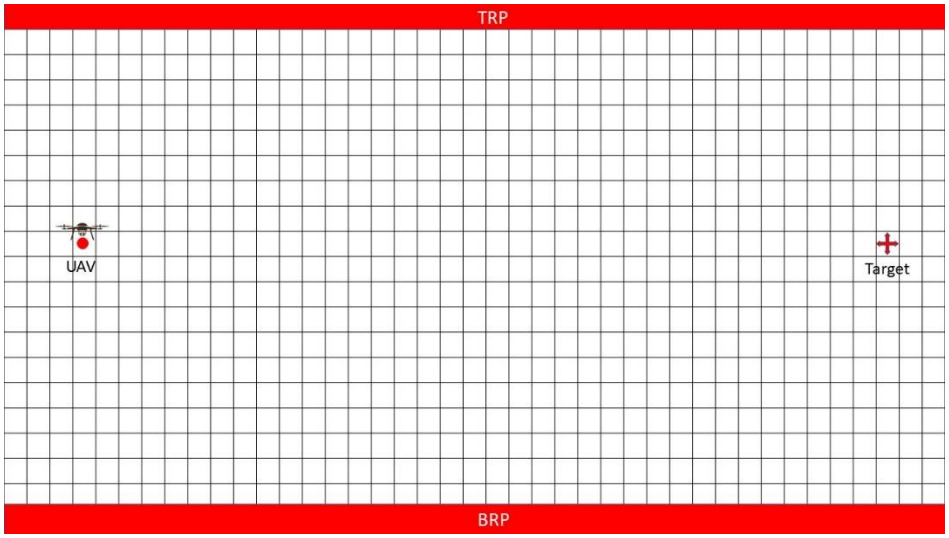


Figure II-8: TEXPLORE application.

II.6.1 Advantages:

- TEXPLORE is a real time approach and it can be deployed on real time systems.
- The agent needs a smaller number of episodes to learn enough about the environment.

II.6.2 Inconvenient:

- The complexity of the system.
- The intensive calculations that take much time.
- Delay of the calculations.

II.7 Conclusion:

Q-learning is a famous AI algorithm and it is used in a lot of domains, and most autonomous UAVs bases on Q-learning algorithm for their behavior controlling. Q-learning can be implemented in different ways depending on the environment and the mission to accomplish. In this chapter we saw the basic Q-learning algorithm and also four applications of autonomous UAVs deploying Q-learning algorithm in different ways and adopting additional algorithms to improve the performance of the UAV, in Table II-1 we can see the different characteristics of the 4 applications.

Applications	Number of UAV	Number of targets	Number of actions	environment	Algorithm
Application 1	1	3	9	3*3 Grid	Q-learning
Application 2	1	1	8	Open area	ARE approach
Application 3	3	clusters	/	Open area	Q-square
Application 4	1	1	8	21*41 Grid	TEXPLOR

Table II-1: A comparison between different applications based on Q-learning.

Chapter III Our Protocol.

III.1 Introduction:

We use a UAV to wirelessly recharge the IoT devices that are located in different locations on the environment, and for the UAV controlling system, we implement a reinforcement learning algorithm named Q-learning to help the UAV control itself without human involvement. So that the UAV learns which action to take at each state by itself and after several episodes the UAV will have a complete knowledge about the environment and the UAV's actions will be more efficient.

III.2 Motivation:

The number of IoT devices is increasing daily and one of the challenges that the IoT devices face is their limited battery life. We consider recharging them, and since it is possible that an IoT device could exist in an isolated place where electricity sources may not exist such as in a mountain, in a middle of a forest, or in middle of a sea. Therefore, we suggest using UAV as flying electrical source to recharge these IoT devices wirelessly, because of its flexible and can fly in any direction with high precision and it can also move faster than a terrestrial vehicle, and since we are using a flying vehicle the environment terrestrial nature is not considered as a challenge to face whether the environment is in middle of sea or on a rocky surface.

With the increasing number of the IoT devices and the changing environment, controlling UAVs by human is not a good solution and it waste a lot of time. Therefore, we implement an algorithm called Q-learning in the UAV controlling system so the UAV does not need to be controlled by human, by implementing this algorithm the UAV learns by itself how to behave and take action in every environment. In our thesis the UAV's main goal is to transfer the maximum amount of energy to the IoT devices, and minimize the number of actions it takes so the transferred energy will be more effective than the energy consumed by the UAV while searching for the IoT devices.

III.3 Description:

One UAV will wirelessly transfer the energy to the four IoT devices that are located in random places on a grid of 3x3 (9 cells), the IoT devices remains in their positions

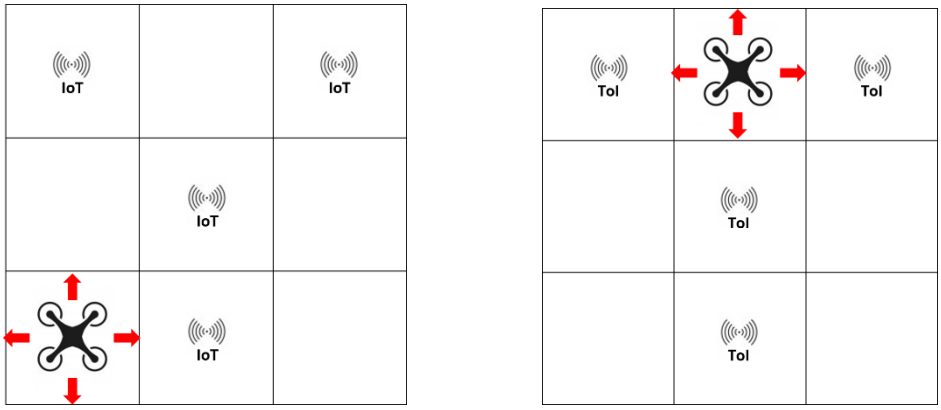


Figure III-1: one UAV and four IoT devices on a Grid of 3x3

(c.f. Figure III-1) and since one of our main goals is to minimize the energy consumed by the UAV the algorithm and the calculations run on a base station, and this base station gives the UAV the actions to take at each step.

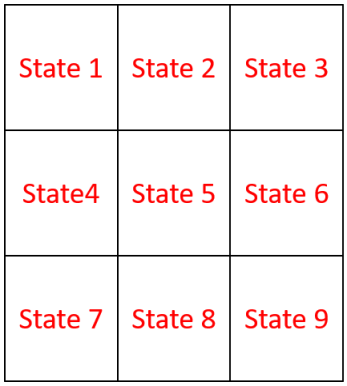
This base station contains the Q-learning algorithm. The Q-learning algorithm is consisted of four main parts, the q-table is where the q-values are stored; a q-function that is used to calculate the q-values; available actions, the UAV is allowed to take one of the four actions at each step; the reward, after taking any action the UAV receives a reward of this specific action and in its state, the reward is used in the q-function.

In the Q-learning process the agent which is the UAV in our contribution runs through episodes and for each episode the UAV takes a number of steps to move around the environment, the UAV may take an action from the four available actions or it may remain in its position in some cases, we set the reward function depending on the behave we want the UAV to follow

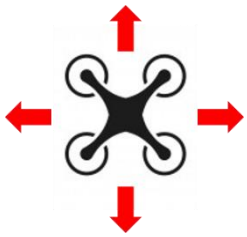
A table named q-table is a size of 9x4 (c.f. Figure III-2.a), 9 is the number of states (c.f. Figure III-2.b) and 4 is the number of actions (c.f. Figure III-2.c) the q-value of each action at each step is saved in this table by the coordinates (state, action).

	up	down	left	right
State 1	0	0	0	0
State 2	0	0	0	0
State 3	0	0	0	0
State 4	0	0	0	0
State 5	0	0	0	0
State 6	0	0	0	0
State 7	0	0	0	0
State 8	0	0	0	0
State 9	0	0	0	0

a) Q-table



b) States



c) Actions

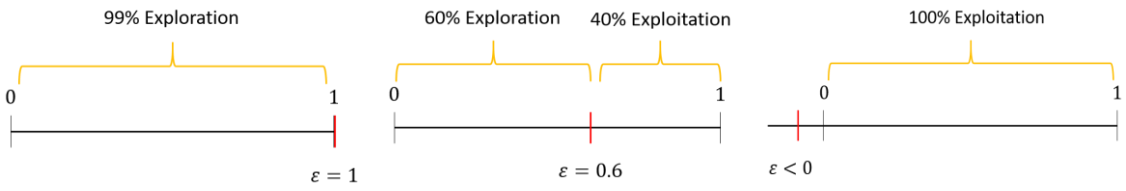
Figure III-2: Application setup.

There are two ways the UAV could pick an action, it may either take a random action from the four available actions or it can take an action depending on the q-table, and the optimal action to take from the q-table at each state is the action with the maximum q-value at this state. These two ways of choosing an action are called Exploration and Exploitation.

At the beginning of the process, we set an exploration rate variable named ε to 1 and after each episode ε decreases by a small decimal number named exploration decay rate, the algorithm at the beginning of the step choses a random decimal number between

0 and 1 than compare it to ε. If this random number is less than ε the algorithm choses the random action way and this way is called Exploration, but if the number is greater than ε then the algorithm choses an action according to the q-table and this way is called Exploitation.

The q-table is initialized by zeroes at the beginning of the process, and since ε is equal to 1 at the first episode and the random number is between 0 and 1, then at each step of the first episode there is a probability of 100% that the algorithm picks a random action and explore the environment (c.f. Figure III-3.a). ε decrease after each episode and the probability of Exploration decreases too on the other side the probability of exploitation increase (c.f. Figure III-3.b), until ε is less than 0 where the algorithm is on 100% Exploitation (c.f. Figure III-3.c), but in our application we set the final value of epsilon



a) Exploration. b) Exploration and Exploitation. c) Exploitation.

Figure III-3:Exploration rate.

to 0.7 because we want the UAV to keep exploring the environment 7% of time even after it learns enough about the environment that means that the final exploitation percentage is 93%.

The rate of changing from 100% Exploration to 93% Exploitation depends on the ε decay rate. For instance, if we set ε decay rate to 0.1 then after 11 episodes ε will be less than 0 then the algorithm will no longer explore the environment and it goes only

with Exploitation. 11 episodes may not be enough for the UAV to explore all the environment especially if it is a vast environment. Therefore, we try to balance between the number of episodes and ε decay rate to give the UAV enough episodes to explore the environment.

While the UAV is learning and exploring the environment some actions may take it out of the environment in some cases, in this situation the UAV will remain in its position and instead of getting a normal reward it will get a penalty for this action to ensure avoiding this action in the next time.

The q-values will be calculated using the Q-function (Equation (1)), and the q-table will be updated at each step. $Q(state_n, action_k)$ is the q-value of the action k in the state n . R is the reward value, α is the learning rate it is a decimal number between 0 and 1 and it controls how much change will be in q-value at each update, high α value means high change, γ is the discounted factor it controls how important the next reward is than the current reward, a high γ value will consider the next reward more important.

III.4 Simulation:

Our code is written in python 3, python is a famous programming language in many fields such as AI, data science, and web development. It a programming language of type script. Our code contains a main class and a main program.

```
import numpy as np
import matplotlib.pyplot as plt
```

we use the library NumPy to manipulate matrices and generate random numbers and for some mathematic functions, and we use the Matplotlib library for plotting the result graphs.

```
class EnvGrid():
    def __init__(self):
        self.grid = [
            [1, 0, 1],
            [0, 1, 0],
            [0, 1, 0]
        ]
        self.y = np.random.randint(0,3)

        self.x = np.random.randint(0,3)
        self.actions = [
            [-1, 0],   # Up
            [0, 1],   # Right
            [1, 0],   # Down
            [0, -1]   # Left
        ]
```

our main class is called **`EnvGrid()`** contains methods, **`__init__()`** is a function used to initialize the variables of the class, in our case we set the **`self.grid`** matrix which represents the environment and **`self.actions`** are the available actions and an initial random location of the UAV on the grid. We use the word self to refer to the main object. **`self.x`** and **`self.y`** are the coordinate of the UAV and each action contains two values, when we add these values to **`self.x`** and **`self.y`**, the coordinates of the UAV on the grid change.

```
    def reset(self):
        self.y = np.random.randint(0,3)
        self.x = np.random.randint(0,3)
        self.energy = 100
        self.te=0
        self.mov=0
        return self.y * 3 + self.x + 1
```

the second method is called **`reset()`**, we use it to reset some variables at the beginning of the episode such as coordinates **`self.x`** and **`self.y`**, **`self.energy`** is

the amount of energy remain in the UAV at the end of the episode, **self.te** is the transferred energy, **self.mov** is the number of the movement per episode. Using the values of **self.x** and **self.y** This method will return the position of the UAV on the q-table.

```
    def step(self, action):
        self.penality = 0
        self.boolean = True
        ycoordinate = self.y + self.actions[action][0]
        xcoordinate = self.x + self.actions[action][1]
        if xcoordinate > 2 or xcoordinate < 0:
            self.boolean = False
        if ycoordinate > 2 or ycoordinate < 0:
            self.boolean = False
        if self.boolean == False:
            self.energy = self.energy - 0.25
            self.penality = -150
        else:
            self.mov+=1
            self.y = self.y + self.actions[action][0]
            self.x = self.x + self.actions[action][1]
            self.energy = self.energy - 0.5
            if self.grid[self.y][self.x] == 1:
                self.energy = self.energy - 0.2
                self.te+=0.2

        reward = (self.grid[self.y][self.x]) + self.penality
        return (self.y * 3 + self.x + 1), reward
```

the third method is called **step (action)**, now after the algorithm pick an action it will take a step using this action, this method will verify if the chosen action will take the UAV out of the environment, and a variable called **self.penalty** will be set to 0 but if the UAV is going to be out of the environment the **self.penalty** variable will be set to -150 and **self.x** and **self.y** will not change, the **self.energy** will be decreased. But if the action will keep the UAV inside the environment the

self.energy will be decreased and the **self.penalty** variable will remain 0 and if the UAV finds an IoT device the **self.energy** will decrease again, this method will return the position of the UAV on the q-table, and the calculated **reward.**

```
    def show(self):
        print("---------------------")
        y = 0
        for line in self.grid:
            x = 0
            for pt in line:
                print("%s\t" % (pt if y != self.y or x != self.x
else "X"), end="")
                x+=1
            y += 1
            print("")
```

the fourth method show is used to print the position of the UAV represented by X on the grid and we can watch how the UAV position will change after each step.

```
    def take_action(st, Q, eps):

        if np.random.uniform(0,1)  < eps:
            action = np.random.randint(0, 4)
        else:
            action = np.argmax(Q[st])
        return action
```

the fifth method **take_action(st, Q, eps)** is used to chosen the action to be token by the UAV; this method takes as parameters **st** which is the current state, **Q** is the q-table of the last update, **eps** is the ε value. **np.random.Uniform(0,1)** is a numpy function to choose a random decimal number between 0 and 1, **np.random.randint(0,4)** is also a numpy function and it is used to pick a random integer between 0 and 3 and we use it to pick a random action, **np.argmax(Q[st])** is a function that returns the index of the action with the maximum q-value.

```
env = EnvGrid ()

Q = [
    [0, 0, 0, 0],
    [0, 0, 0, 0],
    [0, 0, 0, 0],
    [0, 0, 0, 0],
    [0, 0, 0, 0],
    [0, 0, 0, 0],
    [0, 0, 0, 0],
    [0, 0, 0, 0],
    [0, 0, 0, 0],
    [0, 0, 0, 0]
]
num_steps=100
num_episodes=1000
eps = 1
alpha =0.3
discount=0.85
n=0.005
total_reward =[]
energy=[]
transfered_energy=[]
movement=[]
```

Here we created an object using **`EnvGrid ()`** class and we named it env and then we created our Q-table and initialized it with zeros, we also set the number of steps, number of episodes, ε, α, discount factor, n is ε decay rate Table III-1. The rest are empty lists that we will use to store our data to use it later for plotting.

environment	env
Q-table	9*4
Number of steps	100
Number of episodes	1000
ε	1

Learning rate α	0.3
Discount factor γ	0.85
Epsilon decay rate	0.005

Table III-1: Initial parameters

```
for episode in range(num_episodes):

    state = env.reset()
    reward_of_episod = []

    for step in range(num_steps):
        action = take_action(state, Q, eps)
        next_state, r = env.step(action)
        Q[state][action] = Q[state][action] *(1 - alpha ) +
alpha * (r + discount * max(Q[next_state]))

        state = next_state
        reward_of_episod.append(r)

    if eps > 0.07:
        eps -= n
    else:
        eps = 0.07

    transfered_energy.append(env.te/4)
    movement.append(env.mov)
    total_reward.append(sum(reward_of_episod) /
len(reward_of_episod))
    energy.append(env.energy)
```

we use two loops the one that goes through episodes and inside of it there is the second loop that goes through steps, at the beginning of the episode we call the method **`reset ()`** to initialize the location of the UAV and some other variables, we use the list **`reward_of_episod`** to save the rewards of the episode and the end we calculate the mean value of the reward of this episode, and also save the transferred energy to the IoT devices in **`transfered_energy`** and the number of movements in this episode in

a list **movement**, and the amount of energy remains in the UAV battery at the end of the episode in a list **energy**. And then we add the condition part to either decrease epsilon if its greater than 0.7 or set it to 0.7 if its value is less than 0.7 because we want to keep the UAV exploring the environment 7% of time.

The process of each step will be:

- Use the **take_action ()** method and pick an action.
- Use the **step ()** method to take the step.
- Calculate the q-value of the token action.
- Change the current state to the next state.
- Add the reward value to the **reward_of_episod** list.

When the UAV complete the steps loop it will:

- Decrease ε.
- Calculate the average transferred energy and save it in the list **transfered_energy**.
- Save the number of the movement in this episode in **movement**.
- Calculate the mean value of the reward of the episode and save it in the **total_reward** list.
- Save the energy left on the UAV in the list named **energy**.

```
x=np.arange(len(total_reward))
plt.plot(x,total_reward)
plt.ylabel("reward   ")
plt.xlabel("episode")
plt.show()
plt.plot(x, energy)
plt.ylabel("energy   ")
plt.xlabel("episode")
plt.show()
```

```
plt.plot(x, movement)
plt.ylabel("movement")
plt.xlabel("episode")
plt.show()
plt.plot(x, transfered_energy)
plt.ylabel("transferred energy")
plt.xlabel("episode")
plt.show()

for s in range(1, 10):
    print(s, Q[s])
```

when the UAV completes all the episodes and steps, we plot the results using **`plt.plot()`**. **`plot.show()`** is used to show the window of the figure; **`plt.ylabel()`** is used to name the y label and **`plt.xlabel()`** is used to name x label. We created a list of numbers from 0 to the number of episodes and named it x using **`np.arange()`**. And since the all the results we got are per episodes then x is the same for all the plots.

III.5 Results:

After running the code on python editor, we get the results that are presented in four the graphs below.

III.5.1 Reward:

This figure presents the reward of each episode, as we can see that our graph at the beginning starts with low values and after several episodes the reward value increases and converge. That is because the algorithm is changing from exploration to exploitation and when its 93% exploitation that is where the UAV have learned all about the environment and it will only choose the optimal actions, that is why we see the reward converge after about 200 episodes (c.f. Figure III-4).

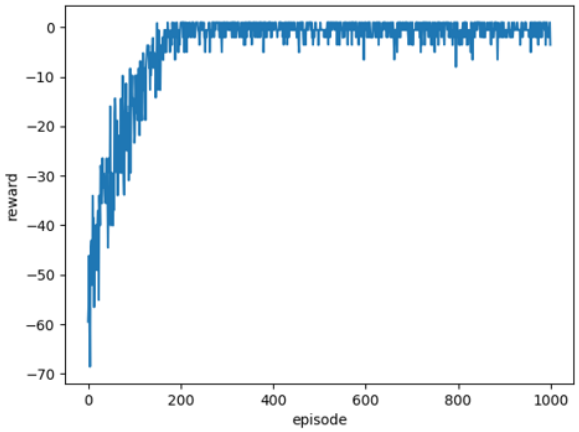


Figure III-4: The reward graph

III.5.2 The energy of the UAV:

This graph presents the energy left on the UAV after each episode, the UAV at the beginning will consume a lot of energy because he still learning and waste a lot of energy moving around but after it learns enough the energy converges to 30% (c.f. Figure III-5).

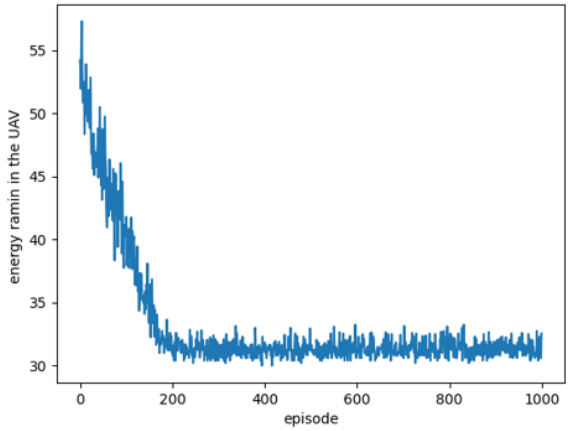


Figure III-5: The energy left on the UAV

III.5.3 The transferred energy to the IoT devices:

as we see the energy transferred to the IoT device is low at the beginning then it increases and that is because the UAV still learning and it does not know the locations of the IoT devices but after it learns enough it will minimize its actions and transfer the most possible amount of the energy (c.f. Figure III-6).

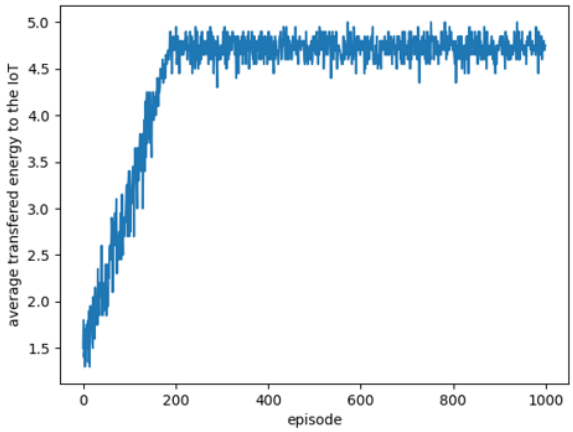


Figure III-6: The transferred energy to the IoT devices

III.5.4 The number of movements at each episode:

The graph of the movement can easily show us when the UAV stops taking penalties and take only the actions that will keep it inside the grid, because the only case the UAV do not move is when it gets a penalty, so at the beginning it will start with minimum number of movement because it still learning and also getting penalties but after learning enough it will no longer take the penalty actions and therefore the number of movements increases (c.f. Figure III-7).

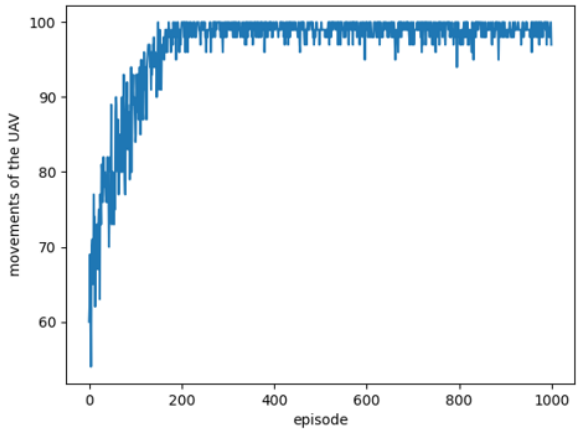


Figure III-7: The number of movements in episodes

III.6 Conclusion:

Our results are enough satisfying for the purpose we want which is maximizing the energy transfer and UAV self-controlling. The Q-learning algorithm worked well for helping the UAV to learn about the environment and helped maximizing the transferred energy to the IoT devices without human involvement, in table (Table III-2) we set the differences between our protocol and the previews protocols and we can see that our protocol have the maximum number of targets and a low number of actions which make it simple and efficient, we also set the advantage and the inconvenient of each protocol.

Applications	Number of UAV	Number of targets	Number of actions	environment	Algorithm	advantage	inconvenient
Application 1	1	3	9	3*3 Grid	Q-learning	Simple approach For energy transfer.	Transferring the energy to all the UAVs all the time.
Application 2	1	1	8	Open area	ARE approach	Considering obstacle avoidance.	Complex approach.
Application 3	3	clusters	/	Open area	Q-square	Multiple UAVs.	Not considering path planning.
Application 4	1	1	8	21*41 Grid	TEXPLOR	Real time path planning.	Complex and longtime approach.
Our application	1	4	4	3*3 Grid	Q-learning	Simple and efficient.	Not transferring the energy equally.

Table III-2: comparison between previews protocols and our protocol.

General conclusion.

RL has become one of the most used AI approaches because of the great satisfying results that it shows specifically in robotics and controlling, and now days machines and robots, and IoT devices exist in an enormous number of domains and they have important roles that replace humans in many fields.

In our thesis work we use a basic RL approach which is Q-learning algorithm and it shows good results, we take in consider the UAV energy and action optimization, and the optimal action value easily converge after a number of episodes. Some point that we did not consider and we can take them as future perspective are defined bellow.

- How to equally transfer the energy to all the IoT devices, even if our algorithm showed good results but the energy transfer between all the IoT devices is still not equal.
- Obstacle avoidance and open environment with large number of users, and we cannot solve this problem with our limited q-table.
- Moving targets, in our case the IoT devices.
- Employing multiple UAVs to keep the process permanently functioning.

REFERENCES

[1] S. Arabi, E. Sabir, H. Elbiaze, and M. Sadik, "Data gathering and energy transfer dilemma in UAV-assisted flying access network for IoT," *Sensors (Switzerland)*, vol. 18, no. 5, pp. 1–20, 2018, doi: 10.3390/s18051519.

[2] S. K. Lee, M. Bae, and H. Kim, "Future of IoT networks: A survey," *Applied Sciences (Switzerland)*, vol. 7, no. 10, pp. 1–25, 2017, doi: 10.3390/app7101072.

[3] Y. Zeng, J. Xu, and R. Zhang, "Energy minimization for wireless communication with rotary-wing UAV," *IEEE Transactions on Wireless Communications*, vol. 18, no. 4, pp. 2329–2345, 2019, doi: 10.1109/TWC.2019.2902559.

[4] A. Knud and L. Lueth, "IoT basics : Getting started with the Internet of Things," *IoT Analytics*, no. March, pp. 0–9, 2015.

[5] Á. Asensio, Á. Marco, R. Blasco, and R. Casas, "Protocol and architecture to bring things into internet of things," *International Journal of Distributed Sensor Networks*, vol. 2014, 2014, doi: 10.1155/2014/158252.

[6] B. Liu, H. Xu, and X. Zhou, "Resource allocation in unmanned aerial vehicle (UAV)-assisted wireless-powered internet of things," *Sensors (Switzerland)*, vol. 19, no. 8, 2019, doi: 10.3390/s19081908.

[7] C. Brooks *et al.*, "A Component Architecture for the Internet of Things," *Proceedings of the IEEE*, vol. 106, no. 9, pp. 1527–1542, 2018, doi: 10.1109/JPROC.2018.2812598.

[8] M. A. Khan *et al.*, "Voting Classifier-based Intrusion Detection for IoT Networks," pp. 1–12, 2021, [Online]. Available: http://arxiv.org/abs/2104.10015.

[9] S. Albishi, B. Soh, A. Ullah, and F. Algarni, "Challenges and Solutions for Applications and Technologies in the Internet of Things," *Procedia Computer Science*, vol. 124, pp. 608–614, 2017, doi: 10.1016/j.procs.2017.12.196.

[10] G. Fersi, "A Distributed and Flexible Architecture for Internet of Things," *Procedia Computer Science*, vol. 73, no. Awict, pp. 130–137, 2015, doi: 10.1016/j.procs.2015.12.058.

[11] S. Dhuli and F. Atik, "Analysis of Distributed Average Consensus Algorithms for Robust IoT networks," pp. 1–8, 2021, [Online]. Available: http://arxiv.org/abs/2104.10407.

[12] C. M. de Morais, D. Sadok, and J. Kelner, "An IoT sensor and scenario survey for data researchers," *Journal of the Brazilian Computer Society*, vol. 25, no. 1, 2019, doi: 10.1186/s13173-019-0085-7.

[13] H. Boyes, B. Hallaq, J. Cunningham, and T. Watson, "The industrial internet of things (IIoT): An analysis framework," *Computers in Industry*, vol. 101, no. December 2017, pp. 1–12, 2018, doi: 10.1016/j.compind.2018.04.015.

[14] A. Anjana, G. Gopi Chand, K. Sai Kiran, J. K. R. Sastry, and Bhpathi, "On improving fault tolerance of iot networks through butterfly networks implemented at services layer," *International Journal of Advanced Trends in Computer Science and Engineering*, vol. 9, no. 2, pp. 2096–2115, 2020, doi: 10.30534/ijatcse/2020/184922020.

[15] V. Hassija, V. Chamola, V. Saxena, D. Jain, P. Goyal, and B. Sikdar, “A Survey on IoT Security: Application Areas, Security Threats, and Solution Architectures,” *IEEE Access*, vol. 7, pp. 82721–82743, 2019, doi: 10.1109/ACCESS.2019.2924045.

[16] H. Nasiri, S. Nasehi, and M. Goudarzi, “Evaluation of distributed stream processing frameworks for IoT applications in Smart Cities,” *Journal of Big Data*, vol. 6, no. 1, 2019, doi: 10.1186/s40537-019-0215-2.

[17] B. Liu and H. Zhu, “Energy-Effective Data Gathering for UAV-Aided Wireless Sensor Networks,” 2019, doi: 10.3390/s19112506.

[18] D. Hodgkinson, R. Johnston, D. Hodgkinson, and R. Johnston, “The future of drones,” *Aviation Law and Drones*, pp. 111–131, 2018, doi: 10.4324/9781351332323-6.

[19] M. Campion, P. Ranganathan, and S. Faruque, “Uav swarm communication and control architectures: A review,” *Journal of Unmanned Vehicle Systems*, vol. 7, no. 2, pp. 93–106, 2019, doi: 10.1139/juvs-2018-0009.

[20] S. I. Granshaw, “RPV, UAV, UAS, RPAS … or just drone?,” *Photogrammetric Record*, vol. 33, no. 162, pp. 160–170, 2018, doi: 10.1111/phor.12244.

[21] H. Chen, Y. Lan, B. K. Fritz, W. Clint Hoffmann, and S. Liu, “Review of agricultural spraying technologies for plant protection using unmanned aerial vehicle (Uav),” *International Journal of Agricultural and Biological Engineering*, vol. 14, no. 1, pp. 38–49, 2021, doi: 10.25165/j.ijabe.20211401.5714.

[22] P. Zhang *et al.*, “Effects of Spray Parameters on the Effective Spray Width of Single-Rotor Drone in Sugarcane Plant Protection,” *Sugar Tech*, vol. 23, no. 2, pp. 308–315, 2021, doi: 10.1007/s12355-020-00890-3.

[23] H. Cheng, J. Page, and J. Olsen, “UAV and obstacle sensing techniques – a perspective,” *International Journal of Intelligent Unmanned Systems*, vol. 1, no. 3, pp. 256–275, 2018, [Online]. Available: https://doi.org/10.1108/IJIUS-01-2013-0001.

[24] T. Elijah, R. S. Jamisola, Z. Tjiparuro, and M. Namoshe, “A review on control and maneuvering of cooperative fixed-wing drones,” *International Journal of Dynamics and Control*, 2020, doi: 10.1007/s40435-020-00710-2.

[25] H. Eisenbeiss, “A mini unmanned aerial vehicle (UAV): system overview and image acquisition,” *International Archives of Photogrammetry. Remote Sensing and Spatial Information Sciences*, vol. 36, no. 5/W1, 2004.

[26] S. H. Alsamhi, O. Ma, M. Samar Ansari, and S. K. Gupta, “Collaboration of drone and internet of public safety things in smart cities: An overview of qos and network performance optimization,” *Drones*, vol. 3, no. 1, pp. 1–18, 2019, doi: 10.3390/drones3010013.

[27] S. Ahsan, R. Naqvi, S. A. Hassan, H. Pervaiz, and Q. Ni, “Naqvi2018.Pdf,” no. January, pp. 36–42, 2018.

[28] J. Torres-Sánchez, F. López-Granados, N. Serrano, O. Arquero, and J. M. Peña, “High-throughput 3-D monitoring of agricultural-tree plantations with Unmanned Aerial

Vehicle (UAV) technology," *PLoS ONE*, vol. 10, no. 6, pp. 1–20, 2015, doi: 10.1371/journal.pone.0130479.

[29] C. Saroglou, P. Asteriou, D. Zekkos, G. Tsiambaos, M. Clark, and J. Manousakis, "UAV-based mapping, back analysis and trajectory modeling of a coseismic rockfall in Lefkada island, Greece," *Natural Hazards and Earth System Sciences*, vol. 18, no. 1, pp. 321–333, 2018, doi: 10.5194/nhess-18-321-2018.

[30] Z. Hu, Z. Bai, Y. Yang, Z. Zheng, K. Bian, and L. Song, "UAV Aided Aerial-Ground IoT for Air Quality Sensing in Smart City: Architecture, Technologies, and Implementation," *IEEE Network*, vol. 33, no. 2, pp. 14–22, 2019, doi: 10.1109/MNET.2019.1800214.

[31] J. Clifton and E. Laber, "Q-Learning : Theory and Applications," pp. 279–303, 2020.

[32] B. Jang and M. Kim, "Q-Learning Algorithms : A Comprehensive Classification and Applications," *IEEE Access*, vol. 7, pp. 133653–133667, 2019, doi: 10.1109/ACCESS.2019.2941229.

[33] L. M. D. da Silva, M. F. Torquato, and M. A. C. Fernandes, "Parallel Implementation of Reinforcement Learning Q-Learning Technique for FPGA," *IEEE Access*, vol. 7, pp. 2782–2798, 2019, doi: 10.1109/ACCESS.2018.2885950.

[34] S. A. Hoseini, J. Hassan, A. Bokani, and S. S. Kanhere, "Trajectory optimization of flying energy sources using Q-learning to recharge hotspot UAVs," *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications Workshops, INFOCOM WKSHPS 2020*, pp. 683–688, 2020, doi: 10.1109/INFOCOMWKSHPS50562.2020.9162834.

[35] Y. Zhao, Z. Zheng, X. Zhang, and Y. Liu, "Q learning algorithm based UAV path learning and obstacle avoidence approach," *Chinese Control Conference, CCC*, pp. 3397–3402, 2017, doi: 10.23919/ChiCC.2017.8027884.

[36] S. Colonnese, F. Cuomo, G. Pagliari, and L. Chiaraviglio, "Q-SQUARE: A Q-learning approach to provide a QoE aware UAV flight path in cellular networks," *Ad Hoc Networks*, vol. 91, p. 101872, 2019, doi: 10.1016/j.adhoc.2019.101872.

[37] N. Imanberdiyev, C. Fu, E. Kayacan, and I. M. Chen, "Autonomous navigation of UAV by using real-time model-based reinforcement learning," *2016 14th International Conference on Control, Automation, Robotics and Vision, ICARCV 2016*, vol. 2016, no. November, pp. 1–6, 2017, doi: 10.1109/ICARCV.2016.7838739.