JOURNAL OF BASIC SCIENCE AND ENGINEERING

Theoretical & mathematical concepts formulation behind the development of AI-ML algorithms for path planning in robots

Sachin R Jadhave [1], Mayura Shelke [2], Dr. Jagdisha N [3], Dr Prashanta G. R[4]

Research Scholar, VTU RRC, Belagavi, Karnataka.
Research Scholar, VTU RRC, Belagavi, Karnataka.
Professor, Department of ISE, Canara Engineering College, Mangalore, India
Professor, Department of CSE, JITD, Davangere, India.

DOI: https://doie.org/10.0824/Jbse.2024438551

Article History: Received: Jan 2022 Revised: April 2024 Accepted: July 2024

Abstract
In this research paper, which is the 1st part of the research article, a brief insight into the theoretical & mathematical modelling concepts of the various types of algorithm development are presented in a concised manner to achieve the desired task, i.e., design of the robot path to move to the destination from the source to the goal in spite of obstacles. Various theoretical concepts of AI & ML algorithms such as the Reinforcement learning, Q-Learning, Deep Learning, Supervised Learning, Un-supervised learning, etc. are being proposed, which are discussed one after the other in a nutshell, first the theoretical concepts are being presented, this is followed by the algorithm development, then the simulated results. The work is divided into a number of sections in the forth coming research papers, where the theoretical concepts of AI & ML presented here are used to develop the hybrid algorithms for the path planning of the robot from the source to the destination. The article is developedin 2 phases, part I & II. The part-I is presented in this research paper, whereas Part II of this research paper is extended to as another research article. The paper concludes with the overall conclusions of the work done in this research article.

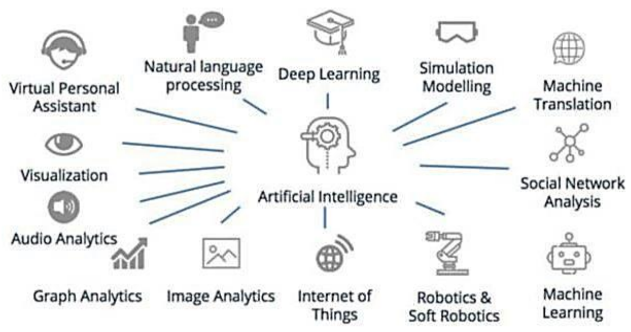
Keywords: AI, ML, DL, Robot, Obstacle, Avoidance, Path, Plan, Goal, Source, Destination, Collision.

Introduction

Artificial Intelligence [AI] is defined as that part of CS & IS with the characteristics with which we associate with intelligence in human behaviour - understanding languages, learning, reasoning, problem solving and so on and is concerned with understanding the nature of intelligent action to perception. It is also defined as the ability of a device to perform functions that are normally associated with human intelligence, such as reasoning, planning, problem solving, pattern recognition, perception, cognition, understanding & learning, possession of sense such as sight, light, sound, touch & feel. The uses of AI are - the collective attributes of a computer, robot or other device capable of performing functions such as learning, decision making, or other intelligent human behaviours [1].

AI is the science & engineering of making intelligent machines. Study of computer systems that attempts to model & apply the intelligence of the human mind is one way of interpreting AI or it can also be termed as the capability of a machine to imitate intelligent human behaviour. AI is composed of 2 words, viz., Artificial & Intelligence. Anything which is notnatural and created by humans is artificial. Intelligence means theability to understand, reason, plan. So any code, technology or algorithm that enables the machine to mimic, develop or demonstrate the human cognition or behaviour is called as AI [2].

The aim of AI is to develop robotic systems that appear to behave intelligently or think like humans. Often this is described in developing robots that think like us. Modern robot control concepts such as the vision, speech, tactile sensing have their roots in AI research. Robot can be considered as a fully automatic machine (an artificial human being) working with Artificial Intelligence. We are the natural human beings working with natural intelligence. The more amount of sensors, memory, learning algos we incorporate, the more intelligent a machine or a robot



becomes. The examples could be the Crays, androids, humanoids, terminators, walking mechanisms, bipeds, unipeds, Asimov, Honda robots and the AIBO's are the robots working with AI. Thus, AI and robotics always go together. The possible applications of AI are shown in the Fig. 1 & 3 [3].

Fig. 1 : Possible applications of AI

The 15 goals of AI & Robotics are – Planning, Problem solving, Natural intelligence, Machine intelligence, Understanding, Cognitive, Learning, Vision, Speech, Reasoning, Fuzzy logic, Behaviour, Expert systems, Neural networks, Programming, Natural Language Processing [4].

ML is an application of AI that provides systems the abilityto automatically learn and improve from experience w/o being explicitly programmed (w/o Human Interactions). Often, this focuses on the development of comp. programs that can access data & use it learn for themselves (similar to humans). ML is an application of AI that provides systems ability to automatically learn & improve from experience. ML enables the computers to tackle tasks that have learnt, until now, only been carried out by people. The primary aim is to allow the computers learn automatically without human intervention or assistance & adjustactions accordingly [5].

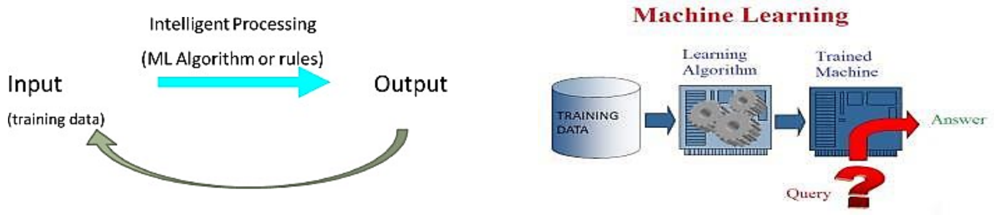


Fig. 2 : Conceptual view of machine learning algorithm

The conceptual view of machine learning algorithm is shown inthe Fig. 2 [6].

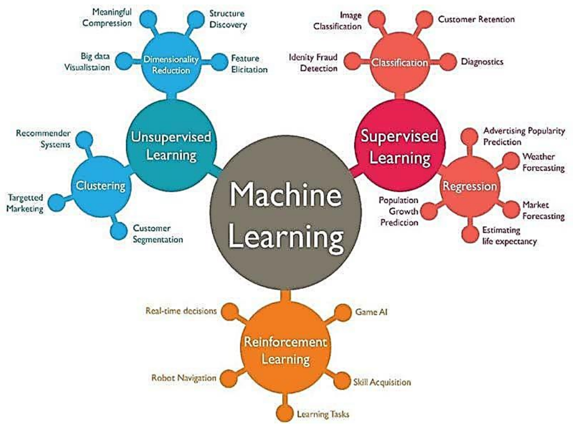


Fig. 3 : Applications of Machine Learning of AI & its types.

The types of ML concepts are graphically shown in the Figs. 4 & 5 respectively (along with the basic types & the abridged ones) [7].

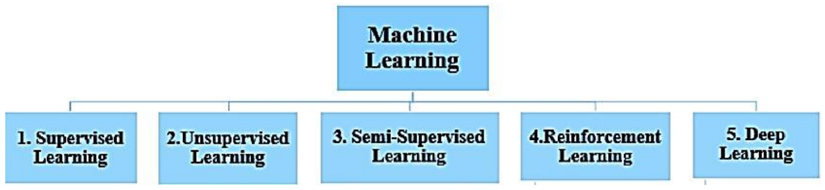


Fig. 4 : Abridged (expanded) types of machine learnings.

Stages of development in machine learning as on date is shown in the Fig. 6. Machine Learning is concerned with computer programs that automatically improve their performance through experience. ML is a field of study that gives the computers the ability to learn without explicitly being programmed as stated by Arthur Samuel in his book. In simple terms, ML means making the predictions based on the datas. A computer program is said to learn from the experiences *E* w.r.t. some class of tasks *T* and performance measure *P* if its performance at tasks in *T*, as measured by *P*, improves with experience *E* is a concised definition of ML as stated by Tom Mitchell. The main advantage of ML are [8]

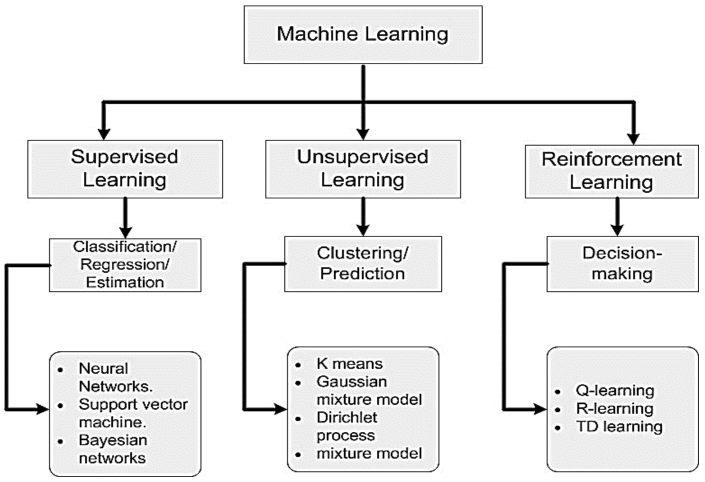


Fig. 5 : Three basic types of machine learning algorithms & its sub divisions

- Learning and writing an algorithm
- It’s easy for human brain but it is tough for a machine & it takes some time and good amount of training data for machine to accurately classify objects Implementation and automation
- This is easy for a machine. Once learnt a machine can process one million images without any fatigue where as human brain can’t & that’s why ML with big data is a deadly combination.

We have seen ML as a buzzword for the past few years, thereason for this is the high amount of data production by applications, the increase of computation power in the past few years and the development of better algorithms for understanding, cognition & recognition [9].

1 – Computer Vision 2 – Imitation Learning

3 – Supervised / Un-Supervised / Reinforcement Learning4 – Assistive and Medical Technologies

5 – Multi-Agent Learning

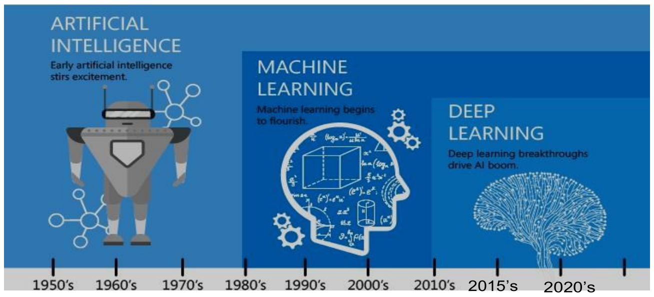

ARTIFICIAL INTELLIGENCE
Early artificial intelligence stirs excitement.
MACHINE LEARNING
Machine learning begins to flourish.
DEEP LEARNING
Deep learning breakthroughs drive AI boom.
1950's
1960's
1970's
1980's
1990's
2000's
2010's
2015's
2020's

Fig. 6 : Stages of development in machine learning as on dateSome examples of ML in use are [10]

Some examples of ML in use are [10]

Prediction — Machine learning can also be used in the prediction systems. Considering the loan example, to compute the probability of a fault, the system will need to classify the available data in groups.
Image recognition — Machine learning can be used for face detection in an image as well. There is a separate category for each person in a database of several people.
Speech Recognition — It is the translation of spoken words intothe text. It is used in voice searches and more. Voice user interfaces include voice dialling, call routing, and appliancecontrol. It can also be used a simple data entry and the preparationof structured documents.
Medical diagnoses — ML is trained to recognize cancerous tissues.
Financial industry and trading — Companies use ML in fraudinvestigations and credit checks.

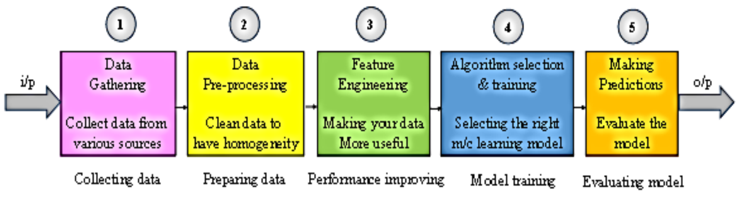


Fig. 7 : Five important steps in any ML algorithm development

The 5 steps to solve a machine learning problem is graphically displayed as shown in the Fig. 7. There are 3 main types of ML algorithms, viz. [11],

- Supervised Learning,
- Unsupervised Learning
- Reinforcement learning

1. Supervised Learning

In Supervised learning, an AI system is presented with data which is labelled, which means that each data tagged with the correct label. The goal is to approximate the mapping function sowell that when you have new input data (x) that you can predict the output variables (Y) for that data. Supervisory learning mechanisms are shown in the Figs. 8 – 10 respectively [12].

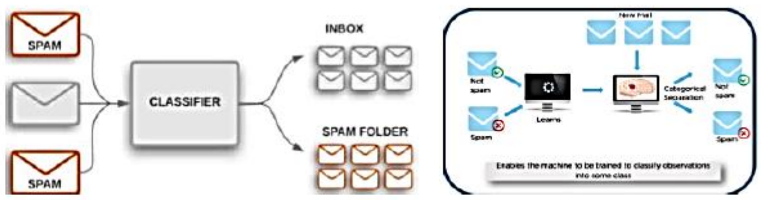
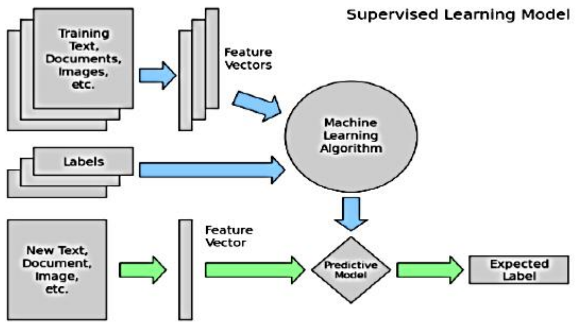


Fig. 8 : Supervisory learning mechanisms

As shown in the above example in Fig. 8, we have initially taken some data and marked them as ‘Spam’ or ‘Not Spam’. Thislabelled data is used by the training supervised model, this data is used to train the model. Once it is trained we can test our modelby testing it with some test new mails and checking of the modelis able to predict the right output. There are 2 types of SL, i.e., Classification - A classification problem is when the output variable is a category, such as “red” or “blue” or “disease” and “no disease” & Regression - A regression problem is when the output variable is a real value, such as “dollars” or “weight” [13].


Supervised Learning Model
Training Text, Documents, Images, etc.
Feature Vectors
Labels
Machine Learning Algorithm
New Text, Document, Image, etc.
Feature Vector
Predictive Model
Expected Label

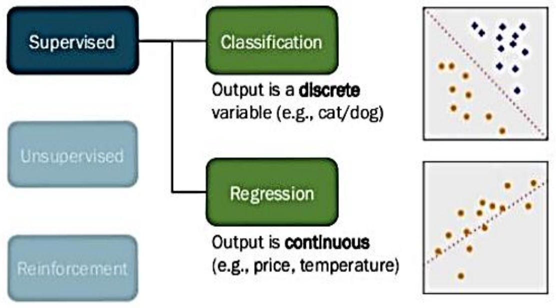

Supervised
Unsupervised
Reinforcement
Classification
Output is a discrete variable (e.g., cat/dog)
Regression
Output is continuous (e.g., price, temperature)

Fig. 9 : Supervised learning model **Fig. 10 : Types of supervised learning**

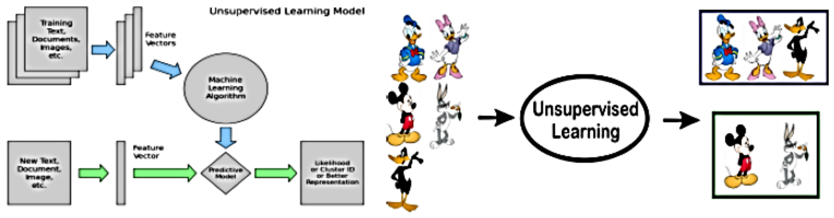

Unsupervised Learning Model
Training Text, Documents, Images, etc.
Feature Vectors
Machine Learning Algorithm
New Text, Document, Image, etc.
Feature Vector
Predictive Model
Likelihood or Cluster ID or Better Representation
Unsupervised Learning

Fig. 11 : Un-supervised learning model & its types

2. Un-Supervised Learning

In unsupervised learning, an AI system is presented with unlabelled, uncategorized data and the system’s algos act on the data without prior training. The output is dependent upon the coded algorithms. Subjecting a system to unsupervised learning is one way of testing AI. Here, we have given some characters toour model which are named as ‘Ducks’ and ‘Not Ducks’. In our training data, we don’t provide any label to the corresponding data. The unsupervised model is able to separate both the characters (ducks & not ducks) by looking at the type of data and models the underlying structure or distribution in the data in order to learn more about it. There is no desired o/p. The types ofUSL are the Clustering - clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behaviour & the second one is the association an association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy X also tend to buy Y. Un-supervised learning model & its types is shown in the Fig. 11 [14].

3. Deep Learning

It is part of a broader family of ML methods based on ANN with representation learning. Learning can be supervised, semi- supervised or unsupervised. Deep learning is a subset of ML in AI that has networks capable of learning unsupervised from datathat is unstructured or unlabelled (deep neural learning or deep neural network). DL is a machine learning technique. It teaches a computer to filter inputs through layers to learn how to predict and classify information. Observations can be in the form of images, text, or sound. The inspiration for DL is the way that thehuman brain filters information [15].

4. Reinforcement learning

Reinforcement Learning is one of the finest branches in Artificial Intelligence. The goal of RL is to enhance an agent’s reward by taking a series of actions of response to a complex environment. Reinforcement Learning is the science which uses experience to make optimal decisions. A reinforcement learning algorithm, or agent, learns by interacting with its environment. The agent receives rewards by performing correctly and penalties for performing incorrectly. The agent learns without intervention from a human by maximizing its reward and minimizing its penalty. It is a type of dynamic programming that trains algorithms using a system of reward and punishment [16].

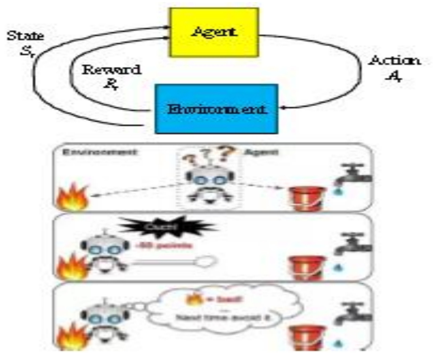


Fig. 12 : Basic components in Reinforcement Learning, agent,environment, reward & action

Here, we can see that the agent is given 2 options, viz., a pathwith water or a path with fire. A reinforcement algorithm works on reward a system, i.e., if the agent uses the fire path then the rewards are subtracted and agent tries to learn that it should avoidthe fire path. If it had chosen the water path or the safe path then some points would have been added to the reward points, the agent then would try to learn what path is safe and what path isn't. It is basically leveraging the rewards obtained, the agent improves its environment knowledge to select the next action. Breaking it down, the Reinforcement Learning process includes these basic steps as shown in the Fig. 12 along with its working in the Fig. 13 [17].

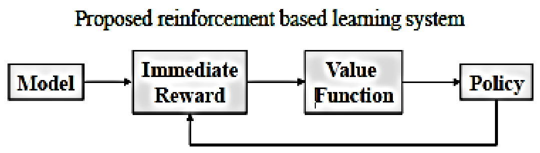


Fig. 13 : Working of Reinforcement learning

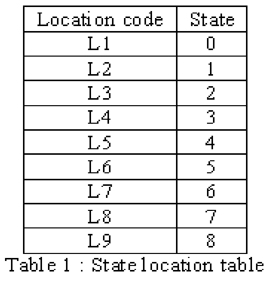
- Observation of the environment.
- Deciding how to act using some strategy.
- Acting accordingly.
- Receiving a reward or penalty.
- Learning from the experiences and refining our strategy.
- Iterate until an optimal strategy is found.

There are 2 main types of algorithms with RL. They are:-

1. model-based
2. model-free

A model-free algorithm is an algorithm which estimates the optimum policy without using or estimating the environmental dynamics (transition and reward functions). Whereas an algorithm based on a model is an algorithm that uses the transition function (and the reward function) to estimate the optimum strategy. It is good to have an established overview of the problem that is to be solved using reinforcement learning, Q-Learning in this case. It helps to define the main components of a reinforcement learning solution i.e. agents, environment, actions, rewards and states. Example for Reinforcement learning& for action taken is shown in the Fig. 14 & 15 respectively [18].

The next, we defining the problem statement as follows. We are to build a few autonomous robots for a guitar buildingfactory. These robots will help the guitar luthiers by conveying them the necessary guitar parts that they would need in order to craft a guitar. These different parts are located at nine different positions within the factory warehouse. Guitars parts include polished wood stick for the fretboard, polished wood for the guitar body, guitar pickups and so on. The luthiers have prioritized the location that contains body woods to be the topmost. They have provided the priorities for other locations as well which we will look in a moment. These locations within the factory warehouse look like [19] –



Location code	State
L1	0
L2	1
L3	2
L4	3
L5	4
L6	5
L7	6
L8	7
L9	8

Table 1 : State location table

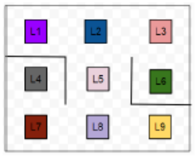


Fig. 14 : Example for Reinforcement learning

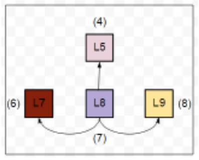


Fig. 15 : Example for action taken in Reinforcement learning

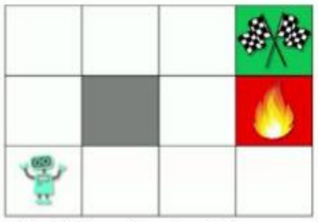


Fig. 16 : Grid search using bellman equation

The actions : In our example, the actions will be the direct locations that a robot can go to from a particular location. Consider, a robot is at the **L8** location and the direct locations to which it can move are **L5**, **L7** and **L9**. The below figure may come in handy in order to visualize this. The set of actions here is nothing but the set of all possible states of the robot. For each location, the set of actions that a robot can take will be different.For example, the set of actions will change if the robot is in L1 [22].

The rewards : By now, we have the following two sets as shownin the rewards table 2 as [23]

- A set of states : S = 0, 1, 2, 3, 4, 5, 6, 7, 8 S = 0, 1, 2, 3, 4, 5, 6, 7, 8
- A set of actions : A = 0, 1, 2, 3, 4, 5, 6, 7, 8

Table 2 : Reward table

Location code	L1	L2	L3	L4L	5L	6L	L7	L8	L9
L1	0	1	0	0	0	0	0	0	0
L2	1	0	1	0	1	0	0	0	0
L3	0	1	0	0	0	1	0	0	0
L4	0	0	0	0	0	0	1	0	0
L5	0	1	0	0	0	0	0	1	0
L6	0	0	1	0	0	0	0	0	0
L7	0	0	0	1	0	0	0	1	0
L8	0	0	0	0	1	0	1	0	1
L9	0	0	0	0	0	0	0	1	0

The rewards, now, will be given to a robot if a location (readit *state*) is directly reachable from a particular location. Let's take an example. L9 is directly reachable from L8. So, if a robot goesfrom L8 to L9 and vice-versa, it will be rewarded by 1. If a location is not directly reachable from a particular location, we do not give any reward (a reward of **0**). Yes, the reward is just a number here and nothing else. It enables the robots to make sense of their movements helping them in deciding what locations are directly reachable and what are not. With this cue, we can construct a reward table which contains all the reward values mapping between all the possible states (locations). The grid search using Bellman's equation is shown in the Fig. 16 [24].

In the above rewards table in table 2, we have all the possiblerewards that a robot can get by moving in between the different states. Now comes an interesting decision. Remember from the previous sections that the luthiers prioritized L6 to be of the topmost? So, how do we incorporate this fact in the above table.This is done by associating the topmost priority location with avery higher reward than the usual ones. Let's put **999** in the cell (**L6**, **L6**): We have now formally defined all the vital componentsfor the solution we are aiming for the problem discussed above. We will shift gears a bit and study some of the fundamental concepts that prevail in the world of reinforcement learning [25].

7. Design of the Bellman's Equation

Bellman equation is the basic block of solving reinforcementlearning and is omnipresent in RL. It helps us to solve MDP. To solve means finding the optimal policy and value functions. The optimal value function $V^*(S)$ is one that yields maximum value. The value of a given state is equal to the max action (action whichmaximizes the value) of the reward of the optimal action in the given state and add a discount factor multiplied by the next state's value from the Bellman Equation [26].

$$V(S) = \max_a \left[R(s,a) + \gamma . V(s') \right]$$

Let's understand this equation, $V(s)$ is the value for being ina certain state. $V(s')$ is the value for being in the next state after taking action a. $R(s, a)$ is the reward we get after taking action ain states. As different actions can be taken, the one with the maximum value is considered because our agent wants to be in the optimal state. γ is the discount factor. The bellman equation in the deterministic environment will be slightly different for a non-deterministic environment or stochastic environment [27].

$$V(s) = \max_a \left(R(s,a) + \gamma \sum_{s'} P(s,a,s') . V(s) \right)$$

In a stochastic environment when we take an action it is notconfirmed that we will end up in a particular next state and thereis a probability of ending in a particular state. $P(s,a,s')$ is the probability of ending is state s' from s by taking action a. This issummed up to a total number of future states. For example, if bytaking an action we can end up in 3 states s_1, s_2, and s_3 from state$s$ with a probability of 0.2, 0.2 and 0.6. The Bellman equation will be [28].

We can solve the Bellman equation using a special technique called dynamic programming. Consider the following square of rooms which is analogous to the actual environment from our original problem but without the barriers. An empty environmentis shown in the table 3 along with the sample environment in thetable 4 followed by the environment with value of the footprintsshown in the table 5 [29].

Now suppose a robot needs to go to the room, marked in green from its current position (A) using the specified direction. How can we enable the robot to do this programmatically? One idea would be to introduce some kind of footprint which the robotwould be able to follow like as shown in the Fig. 17. Here, a constant value is specified in each of the rooms which will comealong the robot's way if it follows the direction specified above.In this way, if it starts at location A, it will be able to scan throughthis constant value and will move accordingly. But this would only work if the direction is prefixed and the robot always starts at location A. Now, consider the robot starts at the following location as shown in the Table 6 @ the position A. Then, the environment with further footprints values can be predicted as shown in the table 7 with the values obtained from the Bellman'sequation depicted in the table 8 [30].

Table 3 : An empty environment

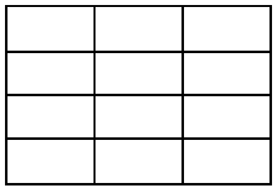
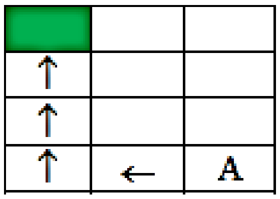
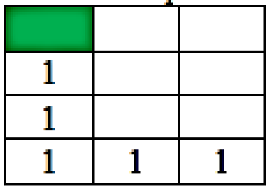


Table 4 : Sample environment



🟩		
↑		
↑		
↑	←	A

Table 5 : Environment with value footprints



🟩		
1		
1		
1	1	1

The robot now sees footprints in two different directions. It is, therefore, unable to decide which way to go in order to get to the destination (green room). It happens primarily because the robot does not have a way to remember the directions to proceed. So, our job now is to enable the robot with a memory. This is where the Bellman Equation comes into play with the mathematical model given by [31].

$$V(S) = \max_a \left[R(s,a) + \gamma . \left(0.2 * V(S_1) + 0.2 * V(S_2) + 0.6 * V(S_3) \right) \right]$$

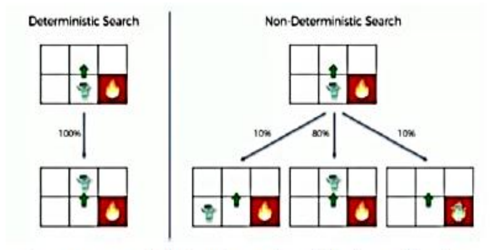


Fig. 17 : Deterministic and non-deterministic search using Bellman equation

🟩	1	0.9
1	0.9	0.81
0.9	0.81	0.729
0.81	0.729	0.66

Table 6 : Environment with further footprint values

🟩		
1		
1		
A	1	1

Table 7 : Environment with further footprint values

🟩		
1		
*		

Table 8 : Environment with footprint values using Bellman equation

🟩		
1		
0.9		
0.81	0.729	Starting point

Fig. 17 : Deterministic and non-deterministic search usingBellman equation

$$V(S)=\max a\left[R(s,a)+\gamma V(s')\right]V(s)=\max a\left(R(s,a)+\gamma V(s')\right)$$

where,

- s = a particular state (room)
- a = action (moving between the rooms)
- s' = state to which the robot goes from s
- γ = discount factor (we will get to it in a moment)
- $R(s,a)$ = a reward function which takes a state s & action a and outputs a reward value
- $V(s)$ = value of being in a particular state (the footprint)

Table values using Bellman equation

We consider all the possible actions and take the one that yields the maximum value. There is one constraint however regarding the value footprint i.e. the room, marked in yellow justbelow the green room will always have a value of 1 to denote thatit is one of the nearest room adjacent to the green room. This is also to ensure that a robot gets a reward when it goes from the yellow room to the green room. Let's now see how to make senseof the above equation here. We will assume a discount factorof 0.9. For this room (read state) what will be $V(s)$? Let's put the values into the equation straightly as [32]

$$V(s)=\max a(0+0.9*1)=0.9V(s)=\max a(0+0.9*1)=0.9$$

Here, the robot will not get any reward for going to the state(room) marked in yellow, hence $R(s , a) = 0$ here. The robot knows the values of being in the yellow room hence $V(s')$ is 1. Following this for the other states, we should get [33]:

A couple of things to notice here:

- The **max()** function helps the robot to always choose the state that gives it the maximum value of being in that state.
- The discount factor ***γ*** notifies the robot about how far it is from the destination. This typically specified by the developer of the algorithm that would be instilled in the robot.

The other states can also be given their respective values in a values obtained from the Bellman's equation as depicted in the similar way with the environmental values with all the footprint table 9 with the graphical display of the deterministic & non- deterministic searches of the best optimized path as shown in the Fig [37].

The robot now can proceed its way through the green room utilizing these value footprints even if it is dropped at anyarbitrary room in the above square. Now, if a robot lands up in the highlighted (sky blue) room, it will still find two options to But eventually, either of the paths will be good enough for the robot to take because of the way the valuefootprints are now laid out [35].

Notice how the main task of reaching a destination from a particular source got broken down to similar subtasks. It turns outthat there is a particular programming paradigm developed especially for solving problems that have repetitive sub problems in them. It is referred to as dynamic programming. It was invented by Richard Bellman in 1954 who also coined the equation we just studied (hence the name, Bellman Equation). Note that this is one of the key equations in the worldof reinforcement learning [1].

If we think realistically, our surroundings do not always work in the way we expect. There is always a bit of stochasticityinvolved in it. This applies to a robot as well. Sometimes, it mightso happen that the robot's inner machinery got corrupted. Sometimes, the robot may come across some hindrances on its way which may not be known to it beforehand. Sometimes, evenif the robot knows that it needs to take the right turn, it will not So, how do we introduce this stochasticity in our case? MarkovDecision Processes [2].

8. Deterministic and non-deterministic search techniques

Deterministic search is the one where the action taken does not involve any randomness for example if the decision taken is to move right then ess. Deterministic search cannot be

$$0.8*V(s_1')+0.1*V(s_2')+0.1*V(s_3')$$

$$V(s)=\max_a\left(R(s,a)+\overbrace{\gamma.V(s')}\right)$$

$$V(s)=\max_a\left(R(s,a)+\gamma\sum_{s'}P(s,a,s').V(s')\right)$$

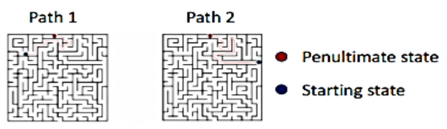


Fig. 18 : Path following using Markov's decision process

used in real time applications which search involves randomness in the actions taken and is most oftenused i deterministicsearch method in our research work. A Markov process is a random process in which the future is independent of the past, given the present. The value of future state depends only on present state but not on how the agent got there (path taken). The entire process could be graphically viewed in the Fig. 17 [3].

Markov Decision Process (MDP)

$$V(s)=\max a\left(R(s,a)+\gamma\left((0.8V(roomup))+0.1V(roomdown)\right)+.....\right)V(s)$$

Markov Decision Process (MDP)

$$=\max a\left(R(s,a)+\gamma\left((0.8V(roomup))+0.1V(roomdown)\right)+.....\right)V(s)+....$$

An important point to remember – each state within a systemis a consequence of its preceding state, which in turn results fromits preceding state. However, processing all of this knowledge would become readily infeasible even for environments with short episodes. Considering, each state follows a Markov property, i.e., each state depends solely on the prior state and the transition from that state to the current state. Let's check out the maze below to better understand how this works [4].

Now, there are 2 scenarios with 2 different starting points, and the agent crosses various paths to reach the same penultimate state. Let us consider the environment with an agent as shown in the Table 10. Then the stochastic environment with an agent could be obtained as shown in the Table 11. Finally, using these 2 previous tables, the environment with an agent (with probabilities) can be computed as shown in the Table 12. Now itdoesn't matter what direction the agent must take to get to the gold [5].

The next step to get out of the labyrinth and attain the last state is to go right. Of course, we just needed the red / penultimatestate knowledge to figure out the next best move which is exactlywhat the Markov property means. Consider the robot is currentlyin the red room and it needs to go to the green room. Let's now consider, the robot has a slight chance of dis-functioning and might take the left or right or bottom turn instead of taking the upper turn in order to get to the green room from where it is now(red room). Now, the question is how do we enable the robot to handle this when it is out there in the above environment? [6]

This is a situation where the decision making regarding which turn is to be taken is partly random and partly under the control of the robot. Partly random because we are not sure when exactly the robot might dysfunction and partly under the controlof the robot because it is still making a decision of taking a turn on its own and with the help of the program embedded into it. Here is the definition of Markov Decision Processes. A Markov decision process (MDP) is a discrete time stochastic control process [7].

$$V(S)=\max a\left[R(s,a)+\gamma V(s')\right]V(s)=\max a\left(R(s,a)+\gamma V(s')\right)$$

It provides a mathematical framework for modelling decision making in situations where outcomes are partly random and partly under the control of a decision maker. We have now introduced ourselves with the concept of partly random and partly controlled decision making. We need to give this concept a mathematical shape (most likely an equation) which then can be taken further. It is surprised to see that we can do this with the help of the Bellman Equation with a few minor tweaks. Here is the original Bellman Equation, again modelled as

What needs to be changed in the above equation so that we can introduce some amount of randomness here? As long as we are not sure when the robot might not take the expected turn, we are then also not sure in which room it might end up in which is nothing but the room it moves from its current room. At this point, according to the above equation, we are not sure of s′ which is the next state (room, as we were referring them). But we do know all the probable turns the robot might take! In order to incorporate each of these probabilities into the above equation, we need to associate a probability with each of the turns to quantify that the robot has got *x* % chance of taking this turn. If we do so, we get [8],

$$V(S) = \max a\left(\left[R(s,a) + \gamma . \sum s' P(s,a,s')\right] V(s')\right) V(s)$$
$$= \max a\left(R(s,a) + \gamma . \sum s' P\ (s,a,s') V(s')\right)$$

Taking the new notations step by step [9]:

- $P(s,a,s')$ - the probability of moving from rooms to rooms′ with action *a*
- $\sum s' P\ (s,a,s') V(s') \sum s' P\ (s,a,s') V(s')$ - expectation of the situation that the robot incurs randomness

Notice, everything else from the above two points is exactly the same. Let's assume the following probabilities are associated with each of the turns the robot might take while being in that red room (to go to the green room). When we associate probabilities to each of these turns, we essentially mean that there is an 80% chance that the robot will take the upper turn. If we put all the required values in our equation, we get [10],

Note that the value footprints will now change due to the fact that we are incorporating stochasticity here. But this time, we will not calculate those value footprints. Instead, we will let the robot to figure it out (more on this in a moment). Up until this point, we have not considered about rewarding the robot for its action of going into a particular room. We are only rewarding the robot when it gets to the destination. Ideally, there should be a reward for every action the robot takes to help it better assess the quality of its actions [11].

The rewards need not be always the same. But it is much better than having some amount reward for the actions than having no rewards at all. This idea is known as the living penalty. In reality, the rewarding system can be very complex and particularly modelling sparse rewards is an active area of research in the domain reinforcement learning. Considering to give a spin to this topic then following resources might come in handy [12]:

- Curiosity-driven Exploration by Self-supervised Prediction.
- Curiosity and Procrastination in Reinforcement Learning.
- Learning to Generalize from Sparse and Under specified Rewards.

In the next section, we will introduce the notion ofthe quality of an action rather than looking at the value of going into a particular room ($V(s)$).

Table 13 : An environment with an agent (with possible valuefootprints)

		🟩
Q (s_4 , a$_4$)} ←	Q (s_1 , a$_1$)} ↑	→ Q (s_2 , a$_2$)}
	O	
	↓ Q (s_3 , a$_3$)}	

8. Transitioning to Q-Learning

By now, we have got the following equation which gives usa value of going to a particular state (*form now on, we will refer to the rooms as states*) taking the stochasticity of the environmentinto the account modelled by [13]
We have also learned very briefly about the idea of living penalty which deals with associating each move of the robot witha reward. Q-Learning poses an idea of assessing the quality of anaction that is taken to move to a state rather than determining thepossible value of the state (value footprint) being moved to. Earlier, we had the environmental values

with an agent with all possible values of the footprints displayed in the table 13 as [14]

If we incorporate the idea of assessing the quality of actions for moving to a certain state **s′**, then the table with the environment belonging to an agent with quality of actions couldbe modified as shown in the table 14 [15].

The robot now has four different states to choose from and along with that, there are four different actions also for the currentstate it is in. So how do we calculate $Q(s\ ,\ a)$ i.e., the cumulative quality of the possible actions the robot might take? Let's breakdown. From the equation as [16]

Table 14 : An environment with an agent (with quality ofactions)

$$V\ (s) = \max a \left(R\ (s,a) + \gamma \sum s'P\ (s,a,s')V\ (s')\right)V\ (s)$$
$$V\ (s) = \max a \left(R\ (s,a) + \gamma \sum s'P\ (s,a,s')V\ (s')\right)$$

		🟩
0.05 $\{V(s_4)\}\leftarrow$	0.8 $\{V(s_1)\}$ $\uparrow$	$\rightarrow$0.05 $\{V(s_2)\}$
	O	
	$\downarrow$ 0.1 $\{V(s_3)\}$	

Note that if we discard the max() function, we get [17],

Essentially, in the equation that produces $V(s)$, we are considering all possible actions and all possible states (from the current state the robot is in) and then we are taking the maximumvalue caused by taking a certain action. The above equation produces a value footprint is for just one possible action. In fact,we can think of it as the quality of the action [18]:

$$Q\left(s,a\right) = R\left(s,a\right) + \gamma \sum s'\left(P\ (s,a,s')V\ (s')\right)Q\left(s,a\right)$$
$$= R\left(s,a\right) + \gamma \sum s'\left(P\ (s,a,s').V\ (s')\right)$$

$$R\left(s,a\right) + \gamma \sum s'P\left(s,a,s'\right)V\left(s'\right) R\left(s,a\right) + \gamma \sum s'P\left(s,a,s'\right)V\left(s'\right)$$

Now that we have got an equation to quantify the quality of a particular action we are going to make a little adjustment in theabove equation. We can now say that $V(s)$ is the maximum of allthe possible values of $Q(s\ ,\ a)$. Let's utilize this fact and replace $V(s')$ as a function of $Q()$ [19]:

$$Q\left(s,a\right) = R\left(s,a\right) + \gamma\ \left(\sum s'\left(P\ (s,a,s')\right)\max a'Q\left(s',a'\right)\right)Q\left(s,a\right)$$
$$= R\left(s,a\right) + \gamma \sum s'\left(P\ (s,a,s').\max a'Q\left(s',a'\right)\right)$$

To ease our calculations. Because now, we have only one function Q() (which is also at the core of the dynamic programming paradigm) to calculate and $R(s\ ,\ a)$ is a quantified metric which produces rewards of moving to a certain state. Thequalities of the actions are called Q-values. And from now on, wewill refer the value footprints as the Q-values [20].

$$V\ (s) = \max a\left(R\ (s,a) + \gamma \sum s'P\ (s,a,s')V\ (s')\right)V\ (s) =$$
$$\left(R\ (s,a) + \gamma \sum s'P\ (s,a,s')V\ (s')\right)$$

Still now, we had discussed about various issues like the AI, ML, DL, RL, SL, USL, Intro to QL, etc… in this 1st part of the research paper. In the 2nd part of the research article, which is the content of another paper, we briefly discuss about the Q Learning Design, Design of a robot navigational path planner using Q- learning, Q-learning algorithm process, Temporal Difference (TD), Deep Q-learning, Deep Q-networks, Neural network and deep Q-learning, Softmax function, Object detection using Histogram of Oriented Gradients with implementation, Path Planners & Path Planning Problem (Gross Motion), Robot Motion Planning [21].

10. Conclusions

In this research paper, a brief insight into the theoretical & mathematical modelling concepts of the various types of algorithm development was presented in a concised manner to achieve the desired task, i.e., design of the robot path to move to the destination from the source to the goal in spite of obstacles viz., the 1st part & the 2nd part, the material presented in this paperis the 1st part of the concepts proposed & the extension of the materials in Part-I is presented as another research paper. Various theoretical concepts of AI & ML algorithms such as the Reinforcement, Q, Deep, Supervised, Unsupervised Learning, etc. are being proposed, which were discussed one after the otherin a nutshell, first the theoretical concepts are being presented. The algorithm work is divided into a number of sections in the forth coming research articles that are going to be published where the theoretical concepts of AI & ML presented here are used to develop the hybrid algorithms for the path planning of the robot from the source to the destination.

REFERENCES

[1]. Zhihao Chen, Redouane Khemmar, Benoit Decoux, Amphani Atahouet, Jean-Yves Ertaud, “Real Time Object Detection, Tracking, and Distance and Motion Estimation based on Deep Learning: Application to Smart Mobility”, *2019 Eighth International Conference on Emerging Security Technologies* (EST), Colchester, United Kingdom, Jul 2019.

[2]. Z. Zhao, P. Zheng, S. Xu and X. Wu, “Object Detection With Deep Learning: A Review”, *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3212-3232, doi: 10.1109/TNNLS.2018.2876865, Nov. 2019.

[3]. L. Jiao et.al., “A Survey of Deep Learning-Based Object Detection”, *IEEE Access*, vol. 7, pp. 128837-128868, 2019.

[4]. Yakup Demir, “Object recognition and detection with deep learning for autonomous driving applications”, Simulation: *Transactions of the Society for Modeling and Simulation International*, pp. 1–11.

[5]. Mukesh Tiwar, Dr. Rakesh Singhai , “A Review of Detection and Tracking of Object from Image and Video Sequences”, *International Journal of Computational Intelligence Research*, ISSN 0973-1873 Vo. 13, No. 5, pp. 745-765, 2017.

[6]. Jahanzaib Shabbir, and TariqueAnwer , “A Survey of Deep Learning Techniques for Mobile Robot Applications”, *Journal of latex class files*, Vol. 14, No. 8, Aug. 2015.

[7]. Bae H., Kim G., Kim J., Qian D., Lee S., “Multi-Robot Path Planning Method Using Reinforcement Learning”, *Journal of Applied Sciences*, vol. 9, issue 15, pp. 30-57, 2019,

[8]. Nguyen, Khang, Huynh, Nhut T., Nguyen, Phat C., Nguyen, Khanh-Duy, Vo Nguyen D., Nguyen, Tam V., “Detecting Objectsfrom Space: An Evaluation of Deep-Learning Modern Approaches”, *Journal of Electronics*, vol. 9, no. 4, pp. 583, 2020.

[9]. Alessandro Foa, “Object Detection in Object Tracking System for Mobile Robot Application”, *Jour. of TRITA-SCI-GRU 2019:090 MAT-E*, 2019:46.

[10]. https://medium.com/@jonathan_hui/ssd-object-detection-single- shot-multibox-detector-for-real-time-processing-9bd8deac0e06

[11]. https://www.educba.com/deep-learning-algorithms

[12]. https://pathmind.com/wiki/deep-reinforcement-learning

[13]. https://towardsdatascience.com/a-beginners-guide-to-q-learning-c3e2a30a653c

[14]. https://www.wikipedia.org

[15]. Dr. T.C.Manjunath, “Fundamentals of Robotics”, 5th Edition, *Nandu Publishers*, *Mumbai*, Inida, 2005.

[16]. Robert, J.S., Fundamentals of Robotics : Analysis and Control, *PHI*, New Delhi., 1992.

[17]. Fu, Gonzalez and Lee, Robotics : Control, Sensing, Vision and Intelligence, *McGraw Hill*, *Singapor*e, 1995.

[18]. Luh, C.S.G., M.W. Walker, and R.P.C. Paul, “On-line computational scheme for mechanical manipulators”, *Journal of Dynamic Systems, Measurement & Control*, Vol. 102, pp. 69-76, 1998.

[19]. Yoshikawa T., “Analysis & Control of Robot Manipulators with Redundancy”, *Proc. First Int. Symp. on Robotics Research, Cambridge, MIT Press*, UK, pp. 735-748, 1984.

[20]. Whitney DE., The Mathematics of Coordinated Control of Prosthetic Arms and Manipulators, *Trans. ASM J. Dynamic Systems, Measurements and Control*, Vol. 122, pp. 303-309, 1972.

[21]. Whitney DE., Resolved Motion Rate Control of Manipulators and Human Prostheses, *IEEE Trans. Syst. Man, Cybernetics*, Vol. MMS-10, No. 2, pp. 47-53, 1969.

[22]. Lovass Nagy V, R.J. Schilling, Control of Kinematically Redundant Robots Using {1}-inverses, *IEEE Trans. Syst. Man, Cybernetics*, Vol. SMC-17, No. 4, pp. 644-649, 1987.

[23]. Lovass Nagy V., R J Miller and D L Powers, An Introduction to the Application of the Simplest Matrix-Generalized Inverse in Systems Science, *IEEE Trans. Circuits and Systems*, Vol. CAS-25, No. 9, pp. 776, 1978.

[24]. D. Xin, C. Hua-hua, and G. Wei-kang, “Neural network and genetic algorithm based global path planning in a static environment,” *Journal of Zhejiang University Science*, vol. 6A, no. 6, pp. 549–554, 2005.

[25]. C. Yang, H. Jianda, and W. Huaiyu, “Quadratic programming- based approch for autonomous vehicle path planning in space,” *Chinese Journal of Mechanical Engineering*, vol. 25, no. 4, pp. 665–673, 2012.

[26]. J.-H. Liang and C.-H. Lee, “Efficient collision-free path-planning of multiple mobile robots system using efficient artificial bee colony algorithm,” *Advances in Engineering Software*, vol. 79, pp.47–56, 2015.

[27]. A. Hidalgo-Paniagua, M. A. Vega-Rodríguez, J. Ferruz, and N. Pavón, “Solving the multi-objective path planning problem in mobile robotics with a firefly-based approach,” *Soft Computing- A Fusion of Foundations, Methodologies and Application*s, vol. 21, no. 4, pp. 949–964, 2017.

[28]. Y. Koren and J. Borenstein, “Potential field methods and their inherent limitations for mobile robot navigation,” *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 1398–1404, Sacramento, CA, USA, April 1991.

[29]. H. Seki, S. Shibayama, Y. Kamiya, and M. Hikizu, “Practical Obstacle Avoidance Using Potential Field for A Nonholonomic Mobile Robot with Rectangular Body,” in *Proceedings of the 13th IEEE International Conference on Emerging Technologies And Factory Automation*, pp. 326–332, Hamburg, Germany, 2008.

[30]. M. Y. Ibrahim and L. McFetridge, “The Agoraphilic algorithm: A new optimistic approach for mobile robot navigation,” *Proc. of the 2001 IEEE/ASME International Conference on Advanced Intelligent Mechatronics Proceedings*, pp. 1334–1339, Como,Italy, July 2001.

[31]. J. Borenstein and Y. Koren, “The vector field histogram—fast obstacle avoidance for mobile robots,” *IEEE Transactions on Robotics and Automation*, vol. 7, no. 3, pp. 278–288, 1991.

[32]. B. You, J. Qui, and D. Li, “A novel obstacle avoidance method for low-cost household mobile robot,” *Proceedings of the 2008 IEEE International Conference on Automation and Logistics* (*ICAL*), pp. 111–116, Qingdao, China, September 2008.

[33]. D. Fox, W. Burgard, and S. Thrun, “The dynamic window approach to collision avoidance,” *IEEE Robotics and AutomationMagazine*, vol. 4, no. 1, pp. 23–33, 1997.

[34]. O. Brock and O. Khatib, “High-speed navigation using the global dynamic window approach,” *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, ICRA99, pp. 341–346, May 1999.

[35]. J. Hong and K. Park, “A new mobile robot navigation using a turning point searching algorithm with the consideration of obstacle avoidance,” *The International Journal of Advanced Manufacturing Technology*, vol. 52, no. 5–8, pp. 763–775, 2011.