Machine Autonomy: Definition, Approaches, Challenges and Research Gaps

Chinedu Pascal Ezenkwu[1] and Andrew Starkey[2]

School of Engineering, University of Aberdeen, Aberdeen, UK
[1]r01cpe17@abdn.ac.uk, [2]a.starkey@abdn.ac.uk

Abstract

The processes that constitute the designs and implementations of AI systems such as self-driving cars, factory robots and so on have been mostly hand-engineered in the sense that the designers aim at giving the robots adequate knowledge of its world. This approach is not always efficient especially when the agent's environment is unknown or too complex to be represented algorithmically. A truly autonomous agent can develop skills to enable it to succeed in such environments without giving it the ontological knowledge of the environment *a priori*. This paper seeks to review different notions of machine autonomy and presents a definition of autonomy and its attributes. The attributes of autonomy as presented in this paper are categorised into low-level and high-level attributes. The low-level attributes are the basic attributes that serve as the separating line between autonomous and other automated systems while the high-level attributes can serve as a taxonomic framework for ranking the degrees of autonomy of any system that has passed the low-level autonomy. The paper reviews some AI techniques as well as popular AI projects that focus on autonomous agent designs in order to identify the challenges of achieving a true autonomous system and suggest possible research directions.

Keywords: Autonomous agent, Machine Autonomy, Automation, Robots, Artificial Intelligence, Learning.

1 Introduction

Autonomous machine intelligence has been a long time concern among roboticists and artificial intelligence researchers; however, there has not been a unifying definition of autonomy in artificial agents. Due to the severalties of its definitions, it is difficult to assess or rate the degree of autonomy of most disruptive and sophisticated systems such as self-driving cars, industrial robots or game playing agents. Importantly, the first step in building a truly autonomous system is the understanding of what it means for a system to be truly autonomous and what the attributes of a system that exhibits autonomy are. Having a coherent conceptual framework would help researchers know how much progress has been made in the syntheses of such systems. Some authors have presented autonomous system in a very broad sense that leaves no distinction between autonomy and automation. For example, Smithers *et al* [1] viewed the autonomous system as the act of building robots. In the same vein, Franklin [2] referred to a remotely controlled mobile robot as an autonomous system. The most popular view is the notion that autonomous robots are to interact with their environments without ongoing human intervention [3]. The use of the word 'ongoing' implies that even if the designer equips the robot *a priori* with all the prevailing knowledge it needs to operate in the environment it would still be labelled autonomous inasmuch as it is not assisted to perform its tasks after deployment.

According to US National Institute of Standards and Technology (NIST) in [4], a system is fully autonomous if it is capable of achieving its goal within a defined scope without human interventions while adapting to operational and environmental conditions. The definition of the autonomous system here is in respect to a well-defined scope. Similarly, Harbers *et al* [5] classified robots, "which are able to perform well-constrained tasks such as surgery, driving on a highway, or vacuum cleaning" as being completely autonomous. It can be argued, however, that such systems are not completely autonomous. Autonomy is when these systems if introduced to an unknown scope or domain, given suitable sensors and actuators for that scope, without changing the algorithm in any way, learns how to handle the sensor signals, adapt their behaviours and act intelligibly. Bradshaw *et al* [6] posits, "since there is no entity that could perform all possible tasks in all possible circumstances; full autonomy does not exist?" This is true even for the most intelligent entity - human. Rich *et al* [7] defines artificial intelligence as "the study of how to make computers do things, which at the moment people do better". This places human-level cognition as the benchmark for building artificial intelligence. Although the computer has outperformed humans in some tasks such as games and some perception tasks, it turns out that our understanding of how humans achieve their high degree of autonomy is paramount in the subject of autonomous systems. Roe *et al* [8] showed that the brain does not require different algorithms to perform different tasks. In their experiment, they rewired the brain of a ferret such that retinal inputs were routed to the auditory pathway. The result demonstrated that the auditory cortex learnt to process visual input after some time. This denotes that the brain is adaptive enough to respond to different forms of sensory data based on a single learning algorithm. In consonance to the preceding, there should be a single algorithm that could enable an autonomous agent to adapt to new situations given suitable sensors without having to teach it how to process and make sense from these sensors' data. To achieve this in any artificial system requires that the system is able to exhibit some attributes of autonomy. This paper seeks to present a definition of machine autonomy and its attributes; and, based on this definition, to provide a review of some underpinning methods and a number of popular researches that aim at autonomous agents. Finally, the paper highlights some research gaps and suggests possible research directions.

2 Attributes of Machine Autonomy

Leveraging on different viewpoints of different researchers on autonomy [9-11], this study considers the attributes of autonomy under two categories – the low-level and the high-level attributes. The low-level attributes are must-have for any autonomous system. They include perception, actuation, learning, context-awareness and decision-making. However, systems that possess only the aforementioned attributes are at the lowest level of autonomy. Conversely, if any of these attributes is missing in any system then such a system cannot be described as autonomous according to these definitions. In contrast, the high-level attributes of autonomy are more advanced attributes and they are the subject of numerous researches in autonomous systems in recent years. They include domain-independence, self-motivation, self-recovery and self-identification of goals. These have been summarised in Table 1 and described in sections 2.1 and 2.2.

Table 1. Attributes of machine autonomy

Low-level Attributes	High-level Attributes
Learning	Domain-independence
Context-awareness	Self-motivation
Actuation	Self-identification of goals
Perception	Self-recoverability
Decision-making	

2.1 Low-level Attributes

Perception. For an agent to make the right decision in its environment it requires information from the environment. Perception is the problem of analysing and representing sensory inputs from dedicated purpose sensors. All autonomous agents must have a means of perceiving their worlds and analysing observations to extract or filter relevant features that would enable them to make sense of the environment. Human agents are naturally equipped with these abilities. However, to enable an artificial agent to exhibit human-level perception is indeed a challenging task. Machine perception has attracted huge research attention. In recent times, there are remarkable contributions in computer vision, natural language processing, feature extraction and dimensionality reduction techniques due to the emergence of some powerful methods in deep learning and machine learning in general.

Actuation. An agent requires a means of providing feedback to the world. Actuation is the ability of an agent to cause a change in both its environment state and/or its internal state. Actuators often "convert other sources of energy such as electric energy, hydraulic fluid or pneumatic pressure to mechanical motion" [12]. The motion is often in response to observations in the agent's environment. Every autonomous agent requires actuators to enable it to act suitably in its environment. Lomonova [12] presents a detailed review of the state-of-the-art actuation systems.

Learning. A learning agent has the ability to make sense from sensory inputs. The science of designing such algorithms is machine learning. Machine learning is an important aspect of computing due to its vast applications across domains. A learning agent can engage in supervised, unsupervised or reinforcement learning depending on the kind of environment it is to operate in and the nature of data that are available to it. A supervised learning agent requires labelled data points to derive a predictive model of its environment. This is different for unsupervised learning technique in which the agent tries to figure out internal structure within unlabelled dataset. The most applied of these three in interactional agent design is the reinforcement learning technique in which an agent makes decisions using unlabelled data points or observations depending on some reward function. Even though learning is one of the necessary attributes of machine autonomy not all learning agents are autonomous.

Context-awareness. Context is an alternative term for the state of an agent's environment. Context-awareness or situational-awareness is the ability of an agent to sense, interpret and adapt to the current context of its environment [13-15]. A context-aware agent has perception and learning abilities and can keep track of a dynamic environment. An autonomous agent should be context-aware i.e. it should have the ability to sense and interpret in real time the prevailing state of its environment and consequently, enabling learning to be contextually relevant.

Decision-making. An important attribute of any computing system is its ability to make decisions. Decision-making is the ability of an agent to map context or perceptual information to action. An intelligent agent should be able to select best actions for all situations. However, decision-making has been implemented in different ways. While some agents depend on hardcoded lookup tables to make their decisions, state-of-the-art techniques are concerned with giving agents the ability to learn optimal and robust decision-making policies. The approach of using hardcoded lookup tables is also referred to as symbolic artificial intelligence or GOFAI (Good Old-Fashioned Artificial Intelligence) [16] while the more recent approach is in non-symbolic artificial intelligence. However, it is difficult to achieve an autonomous GOFAI agent.

2.2 High-level Attributes

Domain-independence. Domain-independent agents do not require the ontological knowledge of their environments at design time to succeed in the environment. This would enable the system to succeed even if the sensors' values change unpredictably [17].The traditional approach of designing the agent whereby the engineer tries to figure out and account for all the possible problems the agent could encounter in the environment, either by hardcoding it or through a task-specific value system, has a huge limitation. The agent would certainly fail woefully if any kind of situation the engineer did not foresee surfaces [18]. A domain-independent algorithm would enable the robot to cope autonomously with any form of environment when given suitable sensors and actuators to sense observations from the environment and act based on the observations using the actuators.

Self-motivation. For an autonomous system to be able to handle an array of interesting tasks as they occur in the environment it should have some level of self-motivation. Typically, intelligent agents are given task-specific knowledge in the form of utility or reinforcement to drive their actions towards achieving a predefined goal. One of the drawbacks of this approach is that the designer must understand the environment and decide the best way to assign values to states and/or actions to enable the agent to act efficiently in the environment. The degree of autonomy of an explicit reward-driven agent would be low since the agent will not be able to adjust its behaviour to handle new interesting states in its environment unless the value system is modified to capture the emerging interests. Assuming a robot is designed to move from a point A to a point B in a navigational environment and the reward function is such that the robot earns a positive reward for avoiding an obstacle and a negative reward for bumping into it. Supposing an unforeseen situation occurs and an obstacle falls on the path to the robot's goal such that the only way this robot can make it to the goal is to displace the obstacle by bumping into it; the robot may not see this as the right thing to do since it is not captured in its reward function. Some researchers have approached this by implementing $\in$-greedy policies which enables the robot to perform some random action with small probability, $\in$. However, a more promising technique is to provide autonomous agents with some degree of self-motivation. Different terms have been used to refer to self-motivation in the literature. Some researchers have viewed self-motivation as intrinsic motivation [19-22], others as artificial curiosity [23-25]while Georgeon et al [26] proposed interactional motivation. However, the key idea of self-motivation is to avoid explicit assignments of values to states or state-action pairs as often done in traditional reinforcement learning. A self-motivated agent is able to act intelligibly in an environment without being hardcoded or being given a task-specific value system. Such an agent uses curiosity to handle situations in the environment, while engaging in an open-ended or lifelong learning [27].

Self-recovery. An interesting application of autonomous agents is the use of robots in environments that are extremely hazardous to humans. The aim of deploying such robots instead of humans in these environments would be defeated if a human technician has to be physically present in the environment to troubleshoot and repair the robots upon failure or reprogram the robot to cope if the environment or the goal changes unpredictably. This necessitates the need for a self-recovery or self-programming mechanisms in autonomous agents. Self-recovery can be proactive, reactive or a fall-back mechanism. In proactive self-recovery, the agent is able to foresee possible causes of failure and devise a solution to abate it. Reactive self-recovery allows the agent to recover from failure after it has occurred. Chaput [18] proposed a fall-back mechanism by which an agent learns different hierarchies of knowledge such that if it encounters a completely strange situation that its higher-level knowledge cannot handle the agent falls back to its lower hierarchy of knowledge and starts building new knowledge that can handle the new situation from there. A self-recovery agent must have the ability to self-analyse itself based on the prevailing situation and build knowledge from the outcome of these analyses. The knowledge gathered in the course of the self-analyses may result in the autonomous reconfiguration of the agent's intelligence mechanism and eventually, an emergent behaviour to suit the situation. Such behaviours may not be explicitly traceable to the agent's program. This has also been referred to as self-programming [28-30]

Self-identification of goal. Goal-oriented agents or robots are the commonest kinds of robots in the literature. Typically, these robots are given goal information either by hardcoding them or by using suitable value systems. Hardcoding goals in an agent program demands that the designer understands, and can formalise, the model of the environment with respect to the agent's actions. This approach will probably fail if this environment is not predictable or known to the designer. For example, it is difficult to hardcode in a self-driving car what constitutes safe driving for every scenario on the road. Furthermore, techniques like the model-free reinforcement learning have successfully applied value systems in numerous applications without the knowledge of transition dynamics of the environment. Nevertheless, it requires prior understanding of the environment and possible actions of the agent in order to be able to design a suitable reward function that would enable the agent to achieve specified goals. To avoid this challenge, there is a need for mechanisms that would enable the agent to self-identify goals and learn suitable skills to enable it to realise them. Simply put, an agent is able to self-identify goals in a given environment if it can develop suitable skills to enable it to achieve a goal that is not explicitly defined in the environment. Although efforts have been made in affordance learning [31-33] and intrinsically motivated learning research [20], self-identification of a goal is still a huge challenge in complex real world tasks.

3 AI Approaches to Autonomous Agents

While autonomy is not possible in symbolic or GOFAI agents, machine learning, evolutionary techniques and developmental artificial intelligence have been widely applied in numerous autonomous agent research. Despite their promising characteristics, none of these techniques has been able to yield a truly autonomous outcome due to some limitations. These

techniques as well as their strengths and weaknesses, in the context of machine autonomy, are discussed in this section. Machine learning is a broad term for techniques that make sense from data. They include supervised, unsupervised and reinforcement learning. Moreover, other techniques have emerged from the diverse ways of implementing the basic machine learning methods. These techniques include active learning and transfer learning. Nonetheless, instead of considering the broad family of machine learning as a topic, these techniques are reviewed individually since they are suitable for different kinds of problems and their strengths and weaknesses are equally different.

3.1 Supervised Learning

A supervised learning agent learns a function that maps input to output given some example of input-output pairs [34,35]. In supervised learning, each data point in the training set is labelled. The learning algorithms analyse the training data and infers a function that can return the corresponding output for a given input. A supervised learning task is solved when the resulting function is able to generalise data points that are not in the training set, otherwise the function is said to overfit. Arrays of solutions have been devised to minimise overfitting during training. These include early stopping [34], cross-validation [36], regularisation [34,37] and dropout technique [38]. Supervised learning algorithms are categorised as classification or regression techniques.

Classification is a form of supervised learning task in which the training samples belong to a finite set of classes. A classifier $f(x)$ is trained to predict the class y, belonging to a finite set of classes, to which an independent input feature vector x belongs [39]. Training the classifier requires a labelled training set, $(x^{(i)}, y^{(i)})_{i=1}^{m}$; where m is the size of the dataset and $(x^{(i)}, y^{(i)})$ is the $i-th$ training example. The label $y^{(i)}$ for all $i \in \{1,2,\dots,m\}$is discrete. Regular classification methods in the literature include k-Nearest Neighbour (KNN), logistic regression, Support Vector Machine (SVM), decision tree, random forest, naïve Bayes and Linear Discriminant Analysis (LDA).

In contrast, regression is the task of approximating a function $f(x)$ from an independent input feature vector x to a continuous output variable, y. Similar to a classifier, training the regressor, $f(x)$,requires a labelled training set, $(x^{(i)}, y^{(i)})_{i=1}^{m}$; where m is the size of the dataset and $(x^{(i)}, y^{(i)})$ is the ith training example. However, the label $y^{(i)}$ for all $i \in \{1,2,\dots,m\}$ is a real-value, such as an integer or floating-point value. Commonly used regression methods include linear regression, Linear Weighted Regression (LWR), Artificial Neural Networks (ANN), ridge regression and Support Vector Regression (SVR).

Some applications of supervised learning in autonomous agent design. Supervised learning has been largely applied in robotics and the development of other AI agents through imitation learning[40,41]. Imitation learning techniques give an agent the ability to learn a policy using a training set obtained from an expert's demonstrations in a similar task [39]. Direct imitation learning is also known as behavioural cloning. Two major problem of direct imitation learning are the correspondence problem [42] and the difficulty for an imitation learning agent to generalise to situations that are not contained in the demonstration dataset. Correspondence problem occurs because of the difference in the architecture, skeleton or degrees of freedom between a human demonstrator and a robot. These two challenges have been better handled using indirect imitation learning or inverse reinforcement learning [42] whereby the agent learns the objectives behind the expert's action in form of a reward function and uses it along with its own experience to improve its behaviours.

The generalisation ability of some supervised learning algorithm such as the deep neural network has helped in improving perceptual tasks in high dimensional feature space. For example, deep learning has helped to improve reinforcement learning in video game playing agents due to its generalisation potential [43] and ability to scale to multi-dimensional feature space [44].

Table 2. Strengths and weaknesses of supervised learning

Strengths	Weaknesses
Supervised learning is a suitable learning technique when a dataset of sensory inputs and the corresponding desirable outputs is available.	Supervised learning algorithms are suitable only when there is a labelled dataset. In the absence of labelled samples, supervised learning cannot be used.
Most supervised learning algorithms have better generalisation than other methods.	Some supervised learning algorithms such as deep learning algorithms require large amounts of data and cannot generalise to simple problems [45].
Supervised learning algorithms such as ANN, SVM with kernel trick, decision tree, random forest and KNN yield non-linear models i.e. they fit into situations where a linear hyperplane is not able to represent the approximation function or the decision boundary for the training samples. This is a huge advantage because real world tasks are often non-linear.	While the knowledge representations in some supervised learning algorithms such as decision trees and naïve Bayes algorithms are transparent, those of more sophisticated and powerful techniques such as ANN, SVM, random forest and deep learning are not interpretable and are considered black box in nature.

3.2 Unsupervised Learning

An unsupervised learning agent learns a pattern in an unlabelled dataset [34]. Unsupervised learning tasks are implemented as clustering techniques, such as k-means, self-organising maps (SOM), adaptive resonance theory (ART), hierarchical models or mixture models. A clustering algorithm is able to automatically figure out the internal structure of a set of inputs without any form of feedback.

Other approaches to unsupervised learning include the Hidden Markov Model (HMM) [46], blind source separation (BSS) [47] and association rule mining [48]. In HMM, the system is assumed to be a Markov process with hidden states while BSS techniques are feature extraction techniques for dimensionality reduction; examples include principal component analysis(PCA), independent component analysis (ICA), non-negative matrix factorization[47] and singular value decomposition. Association rule mining is popular for "discovering interesting relations among variables in a large database" [48]. Common algorithms used for association rule mining include FP-growth algorithm, a priori algorithm and Eclat algorithm [49].

Some applications of unsupervised learning in autonomous agent design. Unsupervised learning algorithms like PCA, SOM, ICA and k-means algorithms are often applied in dimensionality reduction tasks in multimodal sensors tasks(50).Furthermore, some attempts have been made in implementing the forward kinematics of a robot using unsupervised learning algorithms [51,52]. Chaput in [51] implemented a self-recovery mechanism, using a hierarchy of SOMs, which enables a robot to fall back to a lower level of knowledge if its higher-level knowledge cannot handle a situation in the environment.

Table 3. Strengths and weaknesses of unsupervised learning

Strengths	Weaknesses
They are the most suitable approach to pattern recognition when there is no domain knowledge.	It is difficult to decide the correct output or agent action given a set of unlabelled inputs or observations.
They resemble learning in humans and animals[53]more than the supervised learning techniques.	Evaluation of unsupervised learning algorithms is difficult unless there are some labelled samples so that the clustering can be interpreted.
These techniques are often used as data quantisation and dimensionality reduction techniques[54-56] and they have been vastly applied in robotics [57-59].	Some unsupervised learning techniques require some hyperparameters to be chosen a priori before clustering data points. For example, k-means algorithm requires that the number of cluster centres or centroids, k, is chosen correctly.

3.3 Reinforcement Learning

Reinforcement learning differs from the supervised and unsupervised learning in the sense that it is a kind of learning whereby an agent learns an optimal policy for sequential decision-making by interacting with its environment in a trial and error fashion [43,60,61]. A reinforcement learning agent receives "a state s_t, from a state space S, at time t and selects an action a_t from an action space A, following a policy $\pi(a_t|s_t)$, obtains a reward r_t, and transitions to next state s_{t+1}, in accordance with the model of the environment$T(s_{t+1},r_t|s_t,a_t)$" [43,60,61]. The goal of the agent is to maximise expected long-term discounted rewards or the expectation of long-term return over a horizon (episodic or continual). The return is expressed as $R_t = \sum_{k=1}^{\infty} \gamma^k r_{t+k}$, where $\gamma \in (0,1]$ is the discount factor.

For the reinforcement-learning agent to choose the best action a in any state s it must have some function that predicts the measure of how good each state, or state-action pair, is. This function is called the value function. The state value function $v_\pi(s) = E[R|s]$ is the expected return following policy π starting from state s while the action-value function $Q_\pi(s,a) = E[R|s,a,\pi]$ is the expected return of taking action a in state s, then, following policy π. The goal of a reinforcement learning agent is to achieve an optimal policyπ^*. The optimal policy π^* chooses action from $Q^{\pi^*}(s,.)$ or $V^*(s)$ such that the value function is maximised at each state s. Reinforcement learning is the problem of arriving at the optimal policy for a particular task. Assuming the environment dynamics or model is known, two basic approaches to deriving optimal policy are value iteration and policy iteration. This reinforcement learning problem setting is referred to as model-based reinforcement learning. Methods such as the Monte Carlo methods, temporal difference TD and TD(λ) learnings are model-free reinforcement learning. They do not require the environment dynamics to arrive at the optimal policy for any given task. Model-free reinforcement learning is the basis for algorithms like Q-learning [62] and SARSA [60] algorithms which have been widely applied in the literature. To handle generalisation and memory issues reinforcement learning techniques are combined with function approximation such as the least square regression, artificial neural networks, deep learning and so

on. This, especially function approximation with deep learning, has improved the applications of reinforcement learning across domains. In addition, methods that do not care about value function in realising the policy π has equally emerged. These methods are generally called policy gradient methods. An example of a policy gradient method is the REINFORCE algorithm. Likewise, the actor-critic method combines policy iteration, value iteration and policy gradient. This has been widely applied in practice. See [61] for detailed explanations of reinforcement techniques.

Some applications of reinforcement learning in autonomous agent design. Reinforcement learning is one of the most applied techniques in autonomous agent designs due to its ability to allow an agent to improve its behaviour through interactions with the environment. A popular application of reinforcement learning is in the game industry. For example, reinforcement learning was applied in the development of AlphaGo Low, which unlike AlphaGo, the first agent to beat a world Go champion, acquired a superhuman skill in difficult domains, starting *tabula rasa*. AlphaGo Low beat AlphaGo 100-0 in the game of Go [63]. Reinforcement learning has equally been applied in real life robots such as pancake flipping task, bipedal walking energy minimisation task and an archery-based aiming task [64]. Real world applications using *tabula rasa* reinforcement learning can be time consuming and difficult; as such most real world reinforcement learning applications rely on imitation learning and other techniques to hasten the speed of learning.

Table 4. Strengths and weaknesses of reinforcement learning

Strengths	Weaknesses
A reinforcement learning agent does not need labelled dataset like supervised learning methods.	Adequate exploration of the state-action space is required for the agent to have enough experience and knowledge from the environment. This makes reinforcement learning impracticable for most real world tasks.
A reinforcement learning agent is able to learn synchronously through interaction with the environment.	Adequate exploration of the state-action space is required for the agent to have enough experience and knowledge from the environment. This makes reinforcement learning impracticable for most real world tasks.
Model-free reinforcement learning methods do not need the transition dynamics of the environment in order to be effective.	It is difficult to define the reward function for some real world domain [65].
	Sparse and delayed reward can be a problem in several cases.

3.4 Active Learning

An active learning agent tries to learn a model using a dataset of labelled training samples. It estimates how confident the learned model is in predicting the output of a set of the training samples in the unlabelled dataset. The agent queries human experts for the correct outputs of the unlabelled data points which have a low confidence estimate as the case may be. The pool of the labelled dataset is updated with the new input-output pairs and learning continues. Several techniques exist in the literature for implementing active learning [66,67].

Some applications of active learning in autonomous agent design. Active learning is often applied in situations where there are limited annotated demonstration dataset for outright imitation learning. It has been employed to improve imitation learning in 3D navigation tasks [68] in MASH simulator [69]. In [70], active learning for outdoor obstacle detection in a mobile robot using a dataset with severely unbalanced class priors was demonstrated.

Table 5. Strengths and weaknesses of active learning

Strengths	Weaknesses
It fits into situations where there are few labelled data and some unlabelled data.	It requires active participation of a human expert throughout the training and testing process
It is less expensive and less time consuming than supervised learning in situations where an experienced human annotator has to label all training and testing samples.	The agent environment must be known, accessible, understandable and interpretable to the human expert.

3.5 Transfer Learning

Transfer learning is a learning paradigm that allows the use of an already trained model as a starting point to learn a new task. Transfer learning is defined in [71] thus: "Given a source domain D_s and learning task T_s, a target domain D_T and learning task T_T, transfer learning aims to help improve the learning of the target predictive function $f_T(.)$ in D_T using the knowledge in D_s and T_s, where $D_s \neq D_T$, or $T_s \neq T_T$". In the case where $D_s \neq D_T$ holds, then at least one of the following is true: (i.) the domains are of different feature spaces; (ii.) probability distributions of the domains are different. In the case where $T_s \neq T_T$ holds, then at least one of the following is true: (i.) the label spaces are different; (ii.) predictive functions are different. The relevance of transfer learning is that samples are difficult and costly to acquire in most real world problems. Furthermore, it could be more efficient to build knowledge for a new task by tuning an existing knowledge [71]. For example, in [72] the knowledge for solving a simple version of a problem is transferred to a more complex one – transfer learning from 2D to 3D mountain car problem; transfer learning from a Mario game without enemies to a Mario game with enemies. More in-depth reviews of transfer learning techniques are provided in [71,73,74].

Some applications of transfer learning in autonomous agent design. Transfer learning has been applied in [75] to significantly speed up and improve asymptotic performance of reinforcement learning in a physical robot. Kira [76] demonstrated that learning can be successfully sped up between two heterogeneous robots utilizing different sensors and representations by making them transfer support vector machine (SVM) classifiers among each other. Large datasets are often required in several applications of deep convolutional neural networks (CNN). In most cases, knowledge can be transferred from one domain to another by reusing pre-trained convolution layers. In [77], this technique is applied for synthetic aperture radar (SAR) target classification with limited labelled data.

Table 6. Strengths and weaknesses of transfer learning

Strengths	Weaknesses
Transfer learning may speed up learning in a new domain using experiences from a different but somehow similar domain.	Transfer learning may not be an efficient approach when the type of data in the source domain and those from the target domain are too dissimilar.
A transfer learning agent does not require a large amount of experiences in the target domain.	There is no standard way of knowing the size of datasets both in the source and target domains for which transfer learning is suitable.

3.6 Evolutionary Robotics

Evolutionary robotics is a methodology towards autonomous robot development that leverages evolutionary computation, a family of techniques that incorporates principles from biological population genetics to perform search, optimisation, and machine learning [78]. Popular evolutionary computation techniques are genetic algorithms [79], evolutionary strategy [80] and genetic programming [81]. The general idea of evolutionary robotics is to initialise a population of candidate controllers or policies for the robot. After each iteration, the controllers are modified according to a fitness function using some genetic operators such as mutation, crossover and selection of fitter candidates. Fitness function is a metric that reflects the desired performance for the task [78]. A detailed survey and analyses of fitness functions often applied in evolutionary robotics is presented in [82]. A candidate controller may represent a neural network, a collection of rules or a collection of parameter settings [78].

Some applications of evolutionary robotics in autonomous agent design. Evolutionary robotic techniques have been widely applied in some current robotics research [83]. For example, deep neuroevolution, an algorithm that trains a deep neural network using genetic algorithms, has competed with some state-of-the-art algorithms for deep reinforcement learning such as deep Q-network (DQN), asynchronous advantage actor-critic (A3C) and Evolution strategies (ES) in challenging reinforcement learning tasks [84]. The paper reports that deep neuroevolution performed faster than the other three algorithms.

Table 7. Strengths and weaknesses of evolutionary robotics

Strengths	Weaknesses
Evolutionary techniques are adaptive and they are robust to changes in an agent environment.	Evolutionary robotics techniques require fitness function, which is often difficult to craft for most tasks.
Evolutionary robotics is a suitable approach towards autonomous coordinated and cooperative multi-agent systems[85, 86].	Fitness function is always task-specific.

It applies to a wide variety of problems i.e. supervised, unsupervised and reinforcement learning problems.	Training evolutionary algorithms can be computationally intensive depending on the task.

3.7 Developmental Artificial Intelligence

Developmental approach to autonomous systems seeks to replicate infant cognition in artificial systems. This term is often used interchangeably as developmental robotics, autonomous mental development or epigenetic robotics in the literature. In [27] developmental robotics is defined as "an interdisciplinary approach to the autonomous design of behavioural and cognitive capabilities in artificial agents that takes direct inspiration from the developmental principles and mechanisms observed in the natural cognitive system of children". The motivation of developmental artificial intelligence is in line with Turing's idea that "it is easier to build an artificial baby and train it to maturity than trying to build and simulate an adult mind" [87]. In addition to Turing's idea, several works in developmental psychology especially those by Piaget have contributed to the basis for research in developmental artificial intelligence [88,89]. Fundamentally, the developmental artificial intelligence approach seeks to achieve an autonomously open-ended learning driven by intrinsic motivation or artificial curiosity in a way similar to how human infants learn. Given primitive sensors, motors, and a suitable learning algorithm without any prior knowledge of its environment, a developmental agent should be able to bootstrap to maturity out of its own curiosity leading to self-exploration in the environment. Extensive surveys of developmental artificial intelligence are provided in [27, 90-92].

Some applications of developmental artificial intelligence in autonomous agent design. In [93], intelligent adaptive curiosity, an intrinsic motivation mechanism, was used to teach a real robot how to manipulate objects on a baby play mat in such a way that maximises its learning progress. Artificial curiosity has been used to improve reinforcement learning for motion planning on real world humanoids [23].

Table 8. Strengths and weaknesses of developmental artificial intelligence

Strengths	Weaknesses
It makes use of computational models of curiosity, which could enable an agent to explore its environment intelligibly.	It is difficult to train a completely self-motivated agent.
Developmental agents learn in an open-ended manner and are able to adapt to changes in the environment.	Developmental artificial intelligence employs techniques from other methods like reinforcement learning, evolutionary algorithms, etc. and may suffer the inherent weaknesses of these methods.

3.8 Summary

Overall, each of these methods has inherent weaknesses that have made them not suitable for realising full autonomy. However, researchers have adopted synergies of algorithms in an attempt towards developing a true autonomous agent. For example, deep neural networks is combined with reinforcement learning in deep reinforcement learning; deep learning has been combined with evolutionary technique in deep neuroevolution techniques; artificial curiosity as studied in developmental robotics has been integrated with reinforcement learning while active learning has been used with deep imitation learning. Although all of these current methods appear promising, they are still not able to realise full autonomy according to the definition given in this paper. This is largely due to over reliance of these algorithms on externally crafted motivation functions, a human expert experience or domain-specificity of the techniques. The following section presents detailed reviews of popular research papers that have implemented these techniques.

4 Review of Selected Papers

This section considers applications of the techniques or combination of techniques studied in section 3. In order to keep the review as succinct as possible, the studies are summarised in table 10. However, the works are evaluated only on the basis of the high-level attributes desired in the paper. Autonomous system technologies apply to different kinds of systems; nevertheless, the review in this paper considers only embodied and situated agents, either real world robots or simulated agents, which have demonstrated at least all the low-level attributes. The last column of table 10 gives comments for cases to give further details. The papers are reviewed based on the high-level attributes presented in section 2.2. Table 9 recaps the attributes and presents their abbreviations as used in table 10.

Table 9. Meanings of abbreviations used in Table 10

Abbreviation	Meaning	Definition
DI	Domain-independence	Domain-independence is the ability of an agent to cope autonomously with any environment if given suitable sensors and actuators for that environment without being reprogrammed by the designer.
SM	Self-motivation	Self-motivation enables an agent to act intelligibly in an environment without being hardcoded or being given a task-specific value system. Such an agent uses curiosity to handle situations in the environment, while engaging in an open-ended or lifelong learning
SR	Self-recovery	Self-recovery is the ability of an agent to self-analyse and reconfigure itself so as to emerge behaviours that are suitable to the prevailing situations in a changing environment.
SIG	Self-identification of goal	Self-identification of goal is an agent's ability to develop suitable skills to enable it achieve a goal that is not explicitly defined in the environment

Table 10. Summary of related literatures.

Citation	Aim	Method(s)	Contribution	Environment	Attributes demonstrated by the agent				Comment
					DI	SM	SR	SIG	
[68]	To improve the ability of imitation learning to generalise and learn from raw high dimensional data.	Deep imitation learning + Active learning	An approach towards improving generalisation in imitation learning.	Simulated environment	X	X	X	X	The agent was demonstrated in four different environments; however, it completely depends on expert demonstrations to learn how to succeed in each of these environments. This is in contrast to our definition of domain-independence. Other attributes are not demonstrated.
[84]	To demonstrate that a combination of deep learning and genetic algorithms can compete with deep reinforcement learning algorithms in video games	Deep Neuroevolution + Novelty search	A combination of deep neural networks and genetic algorithms as an alternative for deep reinforcement learning. Moreover, the paper demonstrated how novelty search can improve and hasten learning.	Simulated environment	√	√	X	X	The same algorithm was applied in different domains without an account of any change in the algorithm. Self-motivation was implemented as a novelty search, although there was no demonstration on how the agent would perform on the task using only its self-motivation without external reward.
[94]	To implement autonomous mobile robot capable of solving a spiral maze.	Reinforcement learning	Novel concepts called health and sub-health states were suggested	Real world environment	X	X	X	X	The robot's behaviours were entirely dependent on the reward function specified by the experimenter; so, there was no proof of self-motivation. Moreover, there was no attempt to demonstrate other attributes.
[95]	The paper develops a novel strategy that teaches an agent to learn control policies that are suitable for succeeding in a range of Atari 2600 games.	Deep reinforcement learning	A strategy for learning human-level control policies from high dimensional visual input.	Simulated environment	√	X	X	X	It was demonstrated that the agent learnt how to play seven Atari 2600 games with no adjustment of the architecture or learning algorithm. This meets our definition of domain-independence. However, other attributes were not demonstrated.

Citation	Aim	Method(s)	Contribution	Environment	Attributes demonstrated by the agent				Comment
					DI	SM	SR	SIG	
[96]	To demonstrate that intrinsically motivated hierarchical reinforcement learning can enable an artificial agent to learn reusable skills that are needed for competent autonomy	Intrinsic motivation + Hierarchical reinforcement learning	The combination of intrinsic motivation with reinforcement learning	Simulated environment	X	√	X	X	The paper combines both intrinsic (self) and extrinsic (external) rewards or motivations. The intrinsic reward motivated the agent to learn complicated subtasks leading to the goal. The extrinsic motivation is activated after the agent attained the goal state. Furthermore, the paper did not show if the algorithm could adapt to a new environment, or a change in the original environment, without a manual alteration of the agent program or the reward system.
[97]	The develops as strategy that helps to train a robot to be able interact with an unknown environment.	Evolutionary algorithms	A novel chromosome encoding for mobile robotics as well as the simulator for this task was achieved.	Simulated and real world environments	X	X	X	X	A task-specific fitness function was crafted for the agent. The fitness function is not generic enough and would require to be adapted to any new environment if the agent must behave as desired by the designer.
[98]	To implement a curiosity-based intrinsic reward that enables an agent to cope with high dimensional visual inputs.	Intrinsic Curiosity model + reinforcement learning (A3C)	An intrinsic reward signal that "scales to high-dimensional continuous state spaces like images, bypasses the hard problem of predicting pixels and is unaffected by the unpredictable aspects of the environment that do not affect the agent"[98].	Simulated Environments	√	√	X	√	The work in this paper tried to show the possibility of achieving meaningful behaviour using only curiosity as reward. However, the agent progress towards the goal was not yet significant (the agent made a 30% progress in level 1 of the game) but there was a proof of the possibility of purely curiosity-driven learning agent. Moreover, an agent pre-trained at the level 1 of the super Mario bros using only intrinsic motivation as reward performs better in level 2 when fine-tuned than an agent that is trained from the scratch in level 2 using only curiosity.

5 Discussions

Table 10 summarises diverse techniques towards realisations of intelligent systems. However, these methods were implemented in different environments and/or for tasks other than true autonomous systems, as such they cannot be properly compared against each other. To be able to review these approaches for autonomy, the high-level attributes of autonomy, as defined in this paper, were used as the framework for the review. The works in the papers already satisfy all the low-level attributes and as such, using those attributes was not necessary for this review.

Each of these methods professes to be good in the task for which it is designed, however, none of them has demonstrated all the high-level attributes of autonomy. Based on our review, the four best performing methods where adopted for further discussions. These are the deep neuroevolution with novelty search(DNNS)[84] ; deep reinforcement learning(DRL)[95] ; Intrinsically motivated hierarchical reinforcement learning(IMHRL)[96]; and curiosity-driven reinforcement learning(CDRL)[98]. DNNS, DRL and CDRL proved to generalise in more than one environment. However, these environments are closely related in terms of the kind of input signals that the agents receive from the environments and the values/scores the agents try to optimise. Further work is required to ensure that an agent generalises in a variety of significantly unrelated environments.

Although self-motivation was demonstrated by DNNS, IMHRL and CDRL, CDRL demonstrated how much progress an agent can make based on purely intrinsic motivation while other techniques explored the improvements in the agents' behaviours when intrinsic and extrinsic motivations are combined. In DNNS, extrinsic motivation is modelled as a fitness function as is typical with evolutionary techniques while IMHRL used external reward signal for the extrinsic motivation. Using only intrinsic motivation, the CDRL agent was able to make 30% progress in the level 1 of a super Mario game. This was impressive being the first attempt towards a purely curiosity-driven agent in a game environment. However, it requires an improvement.

According to the definition of self-recovery in this paper, none of the techniques demonstrated this attribute. Apart from IMHRL, which was tested in a simple playground environment, the other three were demonstrated in Open AI gym environments. These environments are less flexible to demonstrate self-recoverability of these techniques; as such, a more flexible environment is required for this evaluation.

Amongst the four agents only CDRL demonstrated self-identification of goal. The agent learnt how to navigate the vizdoom environment using only the intrinsic motivation. Similarly, the agent learnt how to make progress while killing the enemies in about 30% of level 1 of the super Mario game. However, there is still room for improvement in more complex and real world environments such as the real world self-driving car or industrial robots in unstructured environments.

6 Conclusion and Research Directions

The paper has reviewed different notions of autonomy and has presented a schema for the classification of autonomy using multidimensional attributes of autonomy. The attributes of machine autonomy presented in the paper are classified into two major groups – low-level and high-level autonomy. The low-level autonomy constitutes the basic attributes that any autonomous system must have while the high-level attributes are bases for evaluating the system's degree of autonomy. It is challenging to think of how best to evaluate any machine's autonomy based on these attributes. Future work should consider creating evaluation metrics for these attributes.

Different AI approaches to autonomous agent design were reviewed and their key challenges highlighted. It is clear that none of the techniques is able to realise all the attributes of autonomy as defined in this paper. Further work is therefore required in developing a single algorithm that would enable an embodied *tabula rasa*, equipped with suitable sensors and actuators, to acquire adequate knowledge to succeed in different environments by adapting and reprogramming itself without any form of human guidance.

Furthermore, most researches in autonomous agents are often experimented in a simulated environment due to high sample complexities of the algorithms that underpin them. This has made it almost impracticable to replicate these algorithms in physical robots despite the ubiquity of high performance computers. Most successful works on physical robots often combine imitation or active learning with other techniques, which are still much dependent on human designers and cannot yield true autonomy. Another reason for this challenge is that robotic simulators for testing autonomy are excessively simple when compared to the real world situations; as such, often times, success in the simulated environment will not imply same in the real world. A way forward could be to develop simulators that present the same difficulty and complexity in training these agents as in the real world while giving the flexibility for researchers to test and evaluate the attributes of autonomy. These can serve as benchmarking tools for research in machine autonomy.

References

1. Smithers T, Laboratory VUBAI. Taking Eliminative Materialism Seriously: A Methodology for Autonomous Systems Research. Artificial Intelligence Laboratory, Vrije Universiteit Brussel (1992).
2. Franklin S. Artificial minds. MIT Press. p.449 (1995).
3. Nolfi S, Floreano D. Evolutionary Robotics: The Biology,Intelligence,and Technology. p. 320, MIT Press (2000).
4. Froese T, Virgo N, and Izquierdo E. Autonomy: A review and a reappraisal. Advances in Artificial Life, pp. 455-464. (2007).
5. Harbers M, Peeters MMM, and Neerincx MA. Perceived Autonomy of Robots: Effects of Appearance and Context. In: Aldinhas Ferreira MI, Silva Sequeira J, Tokhi MO, E. Kadar E and Virk GS eds. A World with Robots: International Conference on Robot Ethics: ICRE 2015. pp. 19-33, Cham: Springer International Publishing, (2017).
6. Bradshaw JM, Hoffman RR, Woods DD, and Johnson M. The seven deadly myths of "autonomous systems". IEEE Intelligent Systems 28: 54-61 (2013).
7. Rich E, Knight K. Artificial Intelligence. McGraw-Hill Higher Education. p.640 (1990).
8. Roe AW, Sur M. Visual projections routed to the auditory pathway in ferrets: Receptive fields of visual neurons in primary auditory cortex. J Neurosci pp. 3651-3664 (1992).
9. Scharre P, Horowitz M. An introduction to autonomy in weapon systems .ethical autonomy—working paper. (2015) .
10. Lane DM. Persistent autonomy artificial intelligence or biomimesis? 2012 IEEE/OES Autonomous Underwater Vehicles (AUV) pp.1-8 (2012).
11. Pernar S. The evolutionary perspective-a transhuman philosophy. (2015).
12. Lomonova EA. Advanced actuation systems — state of the art: Fundamental and applied research. 2010 International Conference on Electrical Machines and Systems pp.13-24 (2010).
13. Hong J, Suh E, and Kim S. Context-aware systems: A literature review and classification. Expert Systems with Applications 36: 8509-8522 (2009).
14. Viterbo J, Sacramento V, Rocha R, Baptista G, Malcher M, and Endler M. A middleware architecture for context-aware and location-based mobile applications. 2008 32nd Annual IEEE Software Engineering Workshop pp.52-61 (2008) .
15. Gui F, Zong N, and Adjouadi M. Artificial intelligence approach of context-awareness architecture for mobile computing. Sixth International Conference on Intelligent Systems Design and Applications 2: 527-533 (2006).
16. Haugeland J. Artificial intelligence: The very idea. cambridge, massachusetts: Bradford. (1985).
17. Pierce D, Kuipers BJ. Map learning with uninterpreted sensors and effectors. Artificial Intelligence 92: 169-227 (1987).
18. Chaput HH. The constructivist learning architecture: A model of cognitive development for robust autonomous robots. PhD. AI Laboratory, The University of Texas at Austin. Supervisors: Kuipers and MiikkulainenAvailable at: https://pdfs.semanticscholar.org/7bb4/18868b2c95443243f5f7a5b9a5a15d342570.pdf; (2004) . Last accessed 2018/08/12
19. Oudeyer PY, Kaplan F, and Hafner VV. Intrinsic motivation systems for autonomous mental development. IEEE Transactions on Evolutionary Computation 11: 265-286 (2007).
20. Barto AG. Intrinsic Motivation and Reinforcement Learning. In: Baldassarre G and Mirolli M eds. Intrinsically Motivated Learning in Natural and Artificial Systems. pp.17-47. Berlin, Heidelberg: Springer Berlin Heidelberg, (2013).
21. Santucci VG, Baldassarre G, and Mirolli M. GRAIL: A goal-discovering robotic architecture for intrinsically-motivated learning. IEEE Transactions on Cognitive and Developmental Systems 8: 214-231 (2016)
22. Schmidhuber J. Formal theory of creativity, fun, and intrinsic motivation (1990–2010). IEEE Transactions on Autonomous Mental Development 2: 230-247 (2010).
23. Frank M, Leitner J, Stollenga M, FÃ¶rster A, and Schmidhuber J. Curiosity driven reinforcement learning for motion planning on humanoids. Frontiers in Neurorobotics pp.7: 25 (2013).
24. Ngo H, Luciw M, Forster A, and Schmidhuber J. Learning skills from play: Artificial curiosity on a katana robot arm. The 2012 International Joint Conference on Neural Networks (IJCNN) pp.1-8 (2012).
25. Gordon G, Ahissar E. A curious emergence of reaching. Advances in Autonomous Robotics. TAROS 2012. Lecture Notes in Computer Science, Berlin, Heidelberg, 7429: 1-12, (2012).
26. Georgeon OL, Marshall JB, and Gay S. Interactional motivation in artificial systems: Between extrinsic and intrinsic motivation. 2012 IEEE International Conference on Development and Learning and Epigenetic Robotics (ICDL) pp.1-2 (2012).
27. Cangelosi A, Schlesinger M. Developmental Robotics: From Babies to Robots. The MIT Press. p.408 (2014).
28. Froese T, Ziemke T. Enactive artificial intelligence: Investigating the systemic organization of life and mind. Artificial Intelligence 173: 466-500 (2009).
29. ThÃ³risson KR, Eric N, Ricardo S, and Pei W. Editorial: Approaches and assumptions of self-programming in achieving artificial general intelligence. Journal of Artificial General Intelligence 3: 1 (2013).
30. Georgeon OL, Marshall JB. Demonstrating sensemaking emergence in artificial agents: A method and an example. Int J Mach Conscious 05: 131-144 (2013).
31. Stramandinoli F, Tikhanoff V, Pattacini U, and Nori F. A bayesian approach towards affordance learning in artificial agents. 2015 Joint IEEE International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob) pp.298-299 (2015).
32. Wang JG, Mahendran PS, and Teoh EK. Deep affordance learning for single- and multiple-instance object detection. TENCON 2017 - 2017 IEEE Region 10 Conference pp.321-326 (2017)
33. Glover AJ, Wyeth GF. Toward lifelong affordance learning using a distributed markov model. IEEE Transactions on Cognitive and Developmental Systems 10: 44-55 (2018).
34. Russell SJ, Norvig P. Artificial Intelligence: A Modern Approach. Pearson Education. pp.1132 (2003).
35. Mohri M, Rostamizadeh A, and Talwalkar A. Foundations of Machine Learning. The MIT Press. p.480 (2012).

36. Ng AY. Preventing "overfitting" of cross-validation data. Proceedings of the Fourteenth International Conference on Machine Learning, pp. 245-253 (1997).
37. Bhlmann P, Geer Svd. Statistics for High-Dimensional Data: Methods, Theory and Applications. Springer Publishing Company, Incorporated. p.573 (2011).
38. Hinton GE, Srivastava N, Krizhevsky A, Sutskever I, and Salakhutdinov R. Improving neural networks by preventing co-adaptation of feature detectors. CoRR 2012; abs/1207.0580 (2012) .
39. Hussein A, Gaber MM, Elyan E, and Jayne C. Imitation learning: A survey of learning methods. ACM Comput.Surv. 50: 21:1-21:35 (2017).
40. Tscherepanow M, Hillebrand M, Hegel F, Wrede B, and Kummert F. Direct imitation of human facial expressions by a user-interface robot. pp.154-160 (2009).
41. Billard A, Grollman D. Imitation Learning in Robots. In: Seel NM ed. Encyclopedia of the Sciences of Learning. pp.1494-1496. Boston, MA: Springer US, (2012).
42. Nehaniv CL, Dautenhahn K. Imitation in Animals and Artifacts. In: Dautenhahn K and Nehaniv CL eds. pp. 41-61. Cambridge, MA, USA: MIT Press, (2002).
43. Szepesvari C. Algorithms for Reinforcement Learning. Morgan and Claypool Publishers. (2010). .
44. LeCun Y, Bengio Y, and Hinton G. Deep learning. Nature pp. 521: 436 (2015).
45. Marcus, G. Deep Learning: A Critical Appraisal, CoRR, **abs/1801.00631** (2018).
46. Lanchantin P, Pieczynski W. Unsupervised restoration of hidden nonstationary markov chains using evidential priors. IEEE Transactions on Signal Processing 53: 3091-3098 (2005).
47. Acharyya R, Ham FM. A new approach for blind separation of convolutive mixtures. 2007 International Joint Conference on Neural Networks, 2075-2080 (2007).
48. Ramezani R, Saraee M, and Nematbakhsh MA. MRAR: Mining multi-relation association rules. Journal of Computing and Security 1(2), (2014) .
49. Heaton J. Comparing dataset characteristics that favor the apriori, eclat or FP-growth frequent itemset mining algorithms. SoutheastCon 2016, pp. 1-7 (2016).
50. Fodor IK. A survey of dimension reduction techniques. Center for Applied Scientific Computing, Lawrence Livermore National Laboratory 9: 1-18. (2002)
51. Chaput HH. The constructivist learning architecture : A model of cognitive development for robust autonomous robots. (2004); .
52. Zhong C, Liu S, Lu Q, and Zhang B. Continuous learning route map for robot navigation using a growing-on-demand self-organizing neural network. International Journal of Advanced Robotic Systems 14: 1729881417743612 (2017).
53. Karhunen, J., Raiko, T. and Cho, K., Unsupervised deep learning: A short review, Available at: https://pdfs.semanticscholar.org/9a9a/9e32ca5cb15e00d0b5a1f2a2656905ba79df.pdf (2014). Last accessed 2018/11/ 24.
54. Jolliffe, I.T. and Cadima, J. Principal component analysis: a review and recent developments, Philosophical Transactions.Series A, Mathematical, Physical, and Engineering Sciences, **374** (2065), pp.20150202 (2016)..
55. Pindah, W., Nordin, S., Seman, A. and Mohamed Said, M.S. Review of dimensionality reduction techniques using clustering algorithm in reconstruction of gene regulatory networks, pp. 172-176 (2015).
56. Xie, H., Li, J. and Xue, H. A survey of dimensionality reduction techniques based on random projection, CoRR, **abs/1706.04371** (2017).
57. Ishii, K., Nishida, S. and Ura, T. A self-organizing map based navigation system for an underwater robot, Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference On, April, pp. 4466-4471 o.5 (2004).
58. Kim, S. and Park, F.C. Fast Robot Motion Generation Using Principal Components: Framework and Algorithms, IEEE Transactions on Industrial Electronics, **55** (6), pp.2506-2516 (2008).
59. Finn, C., Tan, X.Y., Duan, Y., Darrell, T., Levine, S. and Abbeel, P. Learning Visual Feature Spaces for Robotic Manipulation with Deep Spatial Autoencoders, CoRR, **abs/1509.06113** (2015).
60. Sutton RS, Barto AG. Introduction to Reinforcement Learning. Cambridge, MA, USA: MIT Press. (1998) .
61. Sutton RS, Barto AG. Introduction to Reinforcement Learning(2nd Edition, in preparation). MIT Press. (2017).
62. Watkins CJCH, Dayan P. Q-learning. Mach Learning 1992; 8: 279-292.
63. Silver D, Huang A, Maddison CJ, Guez A, Sifre L, van den Driessche G, Schrittwieser J, Antonoglou I, Panneershelvam V, Lanctot M, Dieleman S, Grewe D, Nham J, Kalchbrenner N, Sutskever I, Lillicrap T, Leach M, Kavukcuoglu K, Graepel T, and Hassabis D. Mastering the game of go with deep neural networks and tree search. Nature 529: 484-489 (2016).
64. Kormushev P, Calinon S, and Caldwell GD. Reinforcement learning in robotics: Applications and real-world challenges. Robotics p. 2 (2013) .
65. Abbeel, P. and Ng, A.Y. Apprenticeship learning via inverse reinforcement learning, Proceedings of the Twenty-First International Conference on Machine Learning, Banff, Alberta, Canada, pp. 1 ((2004).
66. Settles B. Active learning literature survey. University of Wisconsin--Madison. (2009).
67. Olsson F. A literature survey of active machine learning in the context of natural language processing. (2009) .
68. Hussein A, Elyan E, Gaber MM, and Jayne C. Deep imitation learning for 3D navigation tasks. Neural Computing and Applications 29: 389-404 (2018).
69. Mash-Simulator. Available at https://github.com/idiap/mash-simulator Last accessed 2018/07/10.
70. Dima C, Hebert M. Active learning for outdoor obstacle detection. (2005) .
71. Pan SJ, Yang Q. A survey on transfer learning. IEEE Trans Knowled Data Eng 22: 1345-1359 (2010).
72. Brys T, Harutyunyan A, Taylor ME, and Now\'e Ann. Policy transfer using reward shaping. pp.181-188 (2015).
73. Weiss K, Khoshgoftaar TM, and Wang D. A survey of transfer learning. Journal of Big Data pp.3: 9 (2016).
74. Tsung F, Zhang K, Cheng L, and Song Z. Statistical transfer learning: A review and some extensions to statistical process control. Quality Engineering 30: 115-128 (2018).
75. Barrett S, Taylor ME, and Stone P. Transfer learning for reinforcement learning on a physical robot. p.1 (2010).

76. Kira Z. Inter-robot transfer learning for perceptual classification. pp.13-20 (2010).
77. Huang Z, Pan Z, and Lei B. Transfer learning with deep convolutional neural network for SAR target classification with limited labeled data. Remote Sensing pp. 9: 907 (2017).
78. Grefenstette JJ. Evolutionary algorithms in robotics†. Fifth International Symposium on Robotics and Manufacturing, ISRAM 94, (1994) .
79. Holland JH. Adaptation in Natural and Artificial Systems. Cambridge, MA, USA: MIT Press. (1992).
80. Vent W. Rechenberg, ingo, evolutionsstrategie — optimierung technischer systeme nach prinzipien der biologischen evolution. 170 S. mit 36 abb. Frommann-Holzboog-Verlag. stuttgart 1973. broschiert. Feddes Repert 86: 337-337 (2008).
81. Luke S, Hamahashi S, and Kitano H. Genetic programming. pp.1098-1105 (1999).
82. Nelson AL, Barlow GJ, and Doitsidis L. Fitness functions in evolutionary robotics: A survey and analysis. Robotics and Autonomous Systems 57: 345-370 (2009).
83. Doncieux S, Bredeche N, Mouret J, and Eiben AEG. Evolutionary robotics: What, why, and where to. Frontiers in Robotics and AI 2: 4 (2015).
84. Such FP, Madhavan V, Conti E, Lehman J, Stanley KO, and Clune J. Deep neuroevolution: Genetic algorithms are a competitive alternative for training deep neural networks for reinforcement learning. CoRR abs/1712.06567, (2017).
85. Duarte, M., Costa, V., Gomes, J.C., Rodrigues, T., Silva, F., Oliveira, S.M. and Christensen, A.L. Evolution of Collective Behaviors for a Real Swarm of Aquatic Surface Robots, CoRR, **abs/1511.03154** (2015).
86. Schrum, J., Lehman, J. and Risi, S. Automatic evolution of multimodal behavior with multi-brain HyperNEAT, pp. 21-22. Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion, Denver, Colorado, USA, (2016).
87. Turing AM. Computers \&Amp; Thought. In: Feigenbaum EA and Feldman J eds. Cambridge, MA, USA: MIT Press, pp.11-35 (1995).
88. Piaget J. The origins of intelligence in children; New York: International Universities Press. (1952).
89. Tsou JY. Genetic epistemology and piaget's philosophy of science: Piaget vs. kuhn on scientific progress. Theory & Psychology 16: 203-224 (2006).
90. Asada M, Hosoda K, Kuniyoshi Y, Ishiguro H, Inui T, Yoshikawa Y, Ogino M, and Yoshida C. Cognitive developmental robotics: A survey. IEEE Transactions on Autonomous Mental Development 1: 12-34 (2009).
91. Guerin F. Learning like a baby: A survey of artificial intelligence approaches. The Knowledge Engineering Review 26: 209-236 (2011).
92. Oudeyer P. Autonomous development and learning in artificial intelligence and robotics: Scaling up deep learning to human-like learning. CoRR. abs/1712.01626 (2017).
93. Oudeyer P, Kaplan F, Hafner VV, and Whyte A. The playground experiment: Task-independent development of a curious robot. In proceedings of AAAI Spring Symposium on Developmental Robotics, pp.42-47 (2005).
94. Zuo B, Chen J, Wang L, and Wang Y. A reinforcement learning based robotic navigation system. IEEE Int. Conf. on Sys., Man, and Cybernetics(SMC), pp. 3452-3457 (2014).
95. Mnih V, Kavukcuoglu K, Silver D, Graves A, Antonoglou I, Wierstra D, and Riedmiller MA. Playing atari with deep reinforcement learning. CoRR. abs/1312.5602 (2013).
96. Singh SP, Barto AG, and Chentanez N. Intrinsically motivated reinforcement learning. In Advances in Neural Information Processing Systems 17 (NIPS). MIT Press (2004) .
97. Assis LdS, Soares AdS, Coelho CJ, and Van Baalen J. An evolutionary algorithm for autonomous robot navigation. Procedia Computer Science 80: 2261-2265. (2016).
98. Pathak D, Agrawal P, Efros AA, and Darrell T. Curiosity-driven exploration by self-supervised prediction. CoRR .abs/1705.05363 (2017).