SSRG International Journal of Computer Science and Engineering
ISSN: 2348 – 8387 /doi:10.14445/23488387/IJCSE-V8I1P102

Volume 8 Issue 1, 5-12, January 2021

Image Sampling Using Q-Learning

Ningxia He[#1]

[#] *Department of Mathematics, Jinan University, Guangzhou, China*

Abstract:

With the advent of the digital information age and multimedia technology development, the amount of image data is increasing day by day. The method of image sampling has been paid much attention to. The traditional triangular mesh sampling method needs to initialize the sampling set and the metric tensor before sampling, which is prone to problems such as unreasonable specification. Therefore, an intelligent image sampling method based on the Q-Learning reinforcement learning algorithm is proposed. Built on the interaction between reinforcement learning agents and the environment, an adaptive sampling method is designed to update agents' characteristics constantly. The experimental results show that this method can achieve the same effect as the traditional triangular mesh sampling method and is more intelligent.

Keywords: *image sampling, reinforcement Learning, Q-Learning algorithm*

I. INTRODUCTION

With the advent of the digital information age and multimedia technology development, the amount of image data is increasing day by day[1-3]. Many practical applications require that the image representation method save the storage space and improve the processing speed and quality of the image[4–7]. To reduce the image's storage space and its transmission bandwidth, it is necessary to sample out the set of detail points from the image and represent the entire image with these sets of points. Therefore, image sampling has become the focus of image processing. A digital image can be represented by two-dimensional matrices. Each matrix grid contains the details of images. To capture the high-level details of images and provide a variable resolution approximation for digital images, some literature presented hierarchical representations, such as quadtrees and pyramids[8–12]. Sullivan and Baker derived a new quadtree construction method using the Lagrange multiplier method to solve the optimal allocation rate without monotone constraints[8]. Shukla proposed a compression algorithm built on tree structure segmentation, making the segmented image optimal[9]. Based on the model, Scholefield and Dragotti proposed an adaptive model based on length penalty, making the model more suitable for image recovery[10]. Burt and Adelson continuously removed the correlation between pixels between the image itself and the image copy, and generate a pyramid data structure by an iterative process, and then achieve the purpose of image compression[11]. The above image compression methods can preserve important features of images, which are of great significance for data storage and transmission.

With the development of image mesh representation[13–16], polygonal meshes show better performances to acquire the details of digital images than regular meshes. In particular, important features such as corners and edges can be caught by the edges of triangular meshes. Adams has proposed a GPRFS framework which is based on the GPR scheme[17]. In his method, some points are supposed to construct a triangle mesh, and then data points are added to the triangle mesh until the approximation error is reached. Different from Adams'method, some methods simplify the triangle meshes, which contain the entire set of sampled data points to obtain the optimal sampling set. For example, Bommes et al. proposed a dynamic node/spring system, which can automatically adjust the points in the boundary region and used a few points to represent the image[18]. Botsch et al. proposed an adaptive mesh generation method based on a metric tensor and used four parameters (minimum and maximum Euclidean edge length, a maximum stretch of measurement, and target length of an edge in measurement) construct the metric tensor[19]. They first select a suitable set of parameter values to get a metric tensor and then adjust the initial meshes and finally get about 80% fewer points than the initial meshes to represent the image. By combining binary spatial partitioning and clustering techniques, Maglo et al. segmented the triangles, which initialize the meshes continuously until the approximation error is satisfied [20]. It not only narrows the sampling points but also preserves the feature of the image. This literature provided better methods for the extraction and representation of image features and improved data storage and transmission performance.

All the above methods are built on the reasonable metric tensor and sampling point set for initial sampling and then optimizing the point set. However, it is also a complex process to develop a set of reasonable metric tensor and sampling point sets in the face of innumerable images. Therefore, it is a developing direction to optimize image sampling points directly. The famous sensor selection problem gives a solution[21]. They selected K optimal sensors by testing all possible numbers $\binom{N}{M}$ with the minimum reconstruction error. But this is a big burden on computers. For example, the possible number of

$\binom{100000}{1000}$ combinations is 10500, which is difficult for the computation. For the sensor selection problem, there are also some relevant methods to solve it, such as branch and bound search [22], convex relaxation [21], heuristic-based methods [23], and so on. These methods have a common drawback, i.e., they can not provide an optimal solution to polynomial time. To deal with the drawback. Basirian et al. proposed a method that samples the graphic signals based on a random walk[24]. The advantage of their work is that it only concerned the local graph information and thus improved sampling effectiveness.

However, Abramenko argued that it might be over-sampled or under-sampled in some environments[25]. Therefore, enhance the quality of intelligent image sampling technology, and automation become the inevitable trend of image sampling technology development. The core of reinforcement learning technology is to find the optimal strategy to win through continuous learning. Reinforcement learning in this aspect has merit. Combined with reinforcement learning theory, improving the image sampling technique of sampling is the principal direction of image sampling technology. To sample the image boundary features, we use reinforcement learning technology to continuously learn and find the optimal strategy to obtain the image features and obtain the image sampling set according to the strategy. Among the algorithms of reinforcement learning, the q-learning algorithm has been widely applied, for example, underwater Wireless Sensor Networks [26], mobile social networks [27], Wireless Networks [28].

This paper's main contribution is that We introduce the reinforcement learning algorithm into the image sampling problem and use the reinforcement learning agent to interact with the environment to realize the features of learning evolution to realize the adaptive sampling of image sampling.

The rest of this paper is organized as follows. In section 2, we propose an image sampling algorithm, which is dealt with in detail. In section 3, the effectiveness of the image sampling algorithm is verified by experiments. Finally, we conclude the whole paper in section 4.

II. MATERIAL AND METHODS

A. Material

a) Introduction of the reinforcement learning

Reinforcement learning has been a hot topic in the field of artificial intelligence in recent years. Reinforcement learning, as a sub-field of machine learning, can interact with the external environment to realize the learning and evolution of agents to have stronger adaptability[29-30]. Figure 1 shows the interaction between the reinforcement learning agent and the environment. The learning evolution of reinforcement learning agents can be summarized as follows:
First: Reinforcement learning agents will detect the environment's state in real-time and perform adjustment action commands to the environment according to their action strategies.
Second: The external environment will change under the agent's action instruction, and the agent will use the form of return function to quantify the advantages and disadvantages of the external environment state change.
Third: Agents optimize and adjust their action strategies according to the value of return function and improve them continuously.

Figure 1. The interactive process between the Action and the environment.

b) Q-Learning algorithm

Reinforcement learning algorithms include model-based and model-free methods. In particular, Q-learning algorithm (1) which use a tow-dimensional table Q about state-action pairs (s, a) to evaluate the advantage and disadvantage of taking action for the particular state is one of the model-free methods, that is, it does not know the probability distribution of the state transition and Reward [31]. After the interaction between the Action and the environment returns the Reward, the q value of the corresponding state-action pair can be updated by the following update formula:

$$Q(s,a) = Q(s,a) + \alpha(r + \gamma \max_a Q(s',a)$$

Where α and γ are the learning rate and discount factor, respectively, both within the range of 0 and 1, q (s, a) and Q(s', a) are action-value functions corresponding to state-action pairs (s, a) and (s,' a), respectively. r represents the immediate Reward.

Algorithm 1 Q-Learning Algorithm

1: **Initialize** Q(s,a) arbitrarily
2: **Repeat**(for each episode):
3: **Initialize** s
4: **Repeat**(for each step of the episode):
5: Choose a from s using policy derived from Q
6: Take Action a, observe r, s.'
7: $Q(s,a) = Q(s,a) + \alpha(r + \gamma \max_a Q(s',a)$
8: s ← s.'
9: **Until** s is terminal
10: **Until** convergence is reached

B. Method

In our sampling scheme based on the q-learning algorithm, starting the RL agent at the position of a pixel with the maximum gradient value interacts with the image through the discrete actions taken by the action-value function observed at the position of the current pixel to obtain the pixel sampling set finally. In this process, the agent tries to obtain a strategy that maximizes the long-term returns (discounted cumulative rewards) [32]. The Markov process (MDP) in the sampling scheme based on the q-learning algorithm is described below.

a) State:

We represent the state as the coordinate of the agent's position on the image corresponding to the pixel. For any position i, the corresponding state s = s(i) is the coordinate formed by the number of rows and columns observed in the two-dimensional image matrix at the corresponding position. For the agent to traverse the entire image, we allow the agent to observe and m × m matrix centered on the corresponding position. In this case, m is the matrix dimension.

b) Action:

We consider a discrete action space A={1, . . . , H}, where H represents the total number of actions. A specific action, a ∈ A, represents the number of steps between the current and new point it+1. The new point it+1 is added to the sampling set M, i.e., M:= M ∪ M {i}. For the specific Action a, the current point has many corresponding new points that can be constituted as a new point set N(it, a). In particular, we randomly choose a new point, it+1, which belongs to the N(it, a) (see figure 2).

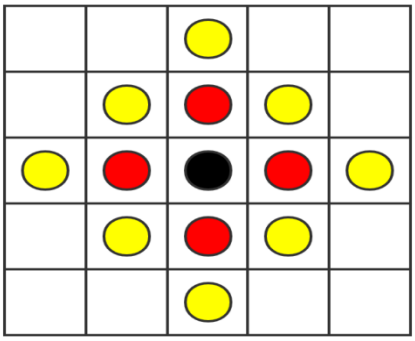


Figure 2. The black point represents the current location at time t. The red points represent the set of the action 1 (N(it, 1)), the yellow points represent the set of the action 2 (N(it, 2)).

In addition, to avoid the agent's tendency to use only, the ε-greedy strategy is constructed to optimize the agent's strategy:

$$\pi^{\varepsilon}(x)=\begin{cases}\pi(x) & \begin{bmatrix}p=1-\varepsilon\\ p=\varepsilon\end{bmatrix}\\ A\end{cases}$$

After optimization, the agent will execute the original strategy with probability 1-ε, and the probability ε will execute any action from the action space A.

c) Reward:

At any point in an image, the agent is intended to select an action that maximizes the discount accumulation reward. Therefore, for agents to move closer to the goal, we should give appropriate Reward to encourage agents to make choices:

$$R := -\min_{\tilde{x}} \|\tilde{x}\|_{TV}$$
$$s.t.\ \tilde{x}[i] = x[i]\ \ for\ all\ i \in M.$$

d) Action-value function:

The point with the maximum gradient value of an image is selected at first. Then the agent draws an action according to the policy π(a|s) in each time step and transforms to the new point it+1, which is selected randomly from the new point set N(it, at). After that, the new point it+1 is put to the sampling set M, i.e., M:=M ∪ M{it+1}. At the same time, we also make an action list, i.e., L:=L ∪ L{at} to record the Action. The process is reiterated until the sampling set M satisfies a prescribed size.

Q-learning's core idea is to make a Q-table of the state and Action, store the action-value Q(s, a), and then select the Action that can get the maximum benefit by the action-value Q(s, a). The updated formula for Q-learning in our method is shown as follows:

$$Q(s,a_t) := \begin{cases} Q(s,a_t)+\alpha*\left(r+\gamma*Q(s,a_{t+1})-Q(s,a_t)\right), & if\ a_t = a_k \\ Q(s,a_t), & if\ a_t = a_k \end{cases}$$

In our work, we make the action-value Q(s, a) update after an episode and not after selecting one Action. As was pointed out earlier, and this is because the Reward is related to all actions which are selected and also because the Reward is observed after the sampling set M is satisfied. The intuition about the update equation(3) is that for each Action that contributes to pick a point into sampling set M (a = at), the action-value Q(s, a) is updated, whereas the Action does not contribute value (a = at), action-value Q(s, a) keep the value the same.

The update of action-value Q(s, a) is repeated with enough episodes to reach the convergence. After that, starting with the maximum gradient value of the point, according to the maximum action-value Q(s, a), we pick a new point into the sampling set. Starting with the new point, we repeat the above procedures until a point does not have suitable Action to select. The above process is summarized in Algorithm 2.

An image can be viewed as a function f which is

defined on a domain Λ. The quality of image approximation is evaluated by the peak signal to noise ratio (PSNR), which is calculated as follows:

$$PSNR = 20\log_{10}\left(\begin{matrix}2^P-1\\ d\end{matrix}\right),\ d=\left(\frac{1}{\Lambda}\sum_{i\in\Lambda}\left|\hat{f}(i)-f(i)\right|^2\right)^{\frac{1}{2}}$$

Where $\hat{f}$ is the image of recovery, and p is the sample precision in bits/sample? The smaller the MSE, the bigger the PSNR. Therefore the bigger the PSNR, the better the image quality.

III. RESULTS

We conducted experiments in three classical data sets and used the proposed sampling method based on q-learning in three different data sets, available from [33] and USCSIPI Image Database [34]. Figure 3 shows two test images, with their respective dimensions (pixels). First, we initialize ε = 0.1, α = 0.1,γ=0.9. And the sampling budget M size is defined by sixty percent of the points of the entire image. We let an agent crawl over the image for selecting the maximum gradient value of point i. According to point i, the agent selects the next point inext using the ε-greedy until the sampling budget M size is satisfied. After that, the Q[s, a] can be restated by the Reward we are learning. This whole process is over until the Q-table is convergent. For each data set, sample points of the whole image are taken according to the action-value function obtained from Table Q, and these points are used to recover the entire image. As shown in figure 4, figure 5.

To verify the effectiveness of our algorithm, using

Algorithm 2 Image point sampling based on Q-Learning algorithm

Input: Image G, sampling budget M, ε, α, γ
Initialize: M={∅ }, L={∅ }, Q-table
Repeat
  select starting point $i \in \Lambda$
  M={i}
  L={∅ }
  for t :=1; t<M do
    a := SampleActi on($\pi(i, a)$)
    i_{next} := SamplePoin t(G, $N(i, a)$)
    $M := M \cup M\{i_{next}\}$
    L := AppendToLi st(L, a)
    $i := i_{next}$
  end for
  $R := -\min_{\tilde{x}} \| \tilde{x} \|_{TV}$
  $s.t.\ \tilde{x}[i] = x[i]$ *for all* $i \in M$.
  for k:=1; k<M do
    for a:=1;a<H do
      $Q(s,a_t) := \begin{cases} Q(s,a_t)+\alpha*(r+\gamma*Q(s,a_{t+1})-Q(s,a_t)), & if\ a_t = a_k \\ Q(s,a_t), & if\ a_t = a_k \end{cases}$
    end for
  end for
Until convergence is reached
Output: Q-table

In the Lena and Peppers images, we compared three point-insertion measurements in a triangular mesh using maximum absolute error (MAE), Laplace maximum error (LMAE), and error standard deviation weighted maximum absolute error (MAES) [35]. For Lena and Peppers' image, these three methods are shown in figure 6, figure 7. By comparing the PSNR values of the three methods (see table 1) for the Lena image, our algorithm can reach the average value of the triangular mesh algorithm. And in the case of the Peppers, our algorithm did a better job of sampling than the other algorithms.

IV. CONCLUSIONS

Image sampling provides crucial technical support for image processing storage, transmission, and compression. Our work proposes a sampling method founded on Q-learning. To effectively and successfully learn with the action-value function, we use the ε-greedy strategic learning method, which assigns a large selection probability to the actions of the action-value function, while the other actions assign the same small selection probability. The agent is utilized to interact with the image, and the corresponding action-value function is updated after a period of time when the Reward is obtained. Compared with the triangular mesh method, we use the Q table after convergence to directly sample the image pixel points instead of initializing the triangular mesh and optimizing sampling points. Through the comparison of PSNR values, compared with the triangular mesh method, our algorithm not only achieves the effect of the traditional triangular mesh algorithm but also realizes a more intelligent effect.

(a) Lena (256 × 256 pixels)

(b) Peppers (256 × 256 pixels)

Figure 3. Two data sets used in the experiments

(a) sample points

(b) recovery of image Lena

Figure4. Representation of image Lena: (a) sample points; (b) recovery of image Lena, PSNR = 24.71

(a) sample points

(b) recovery of image Peppers

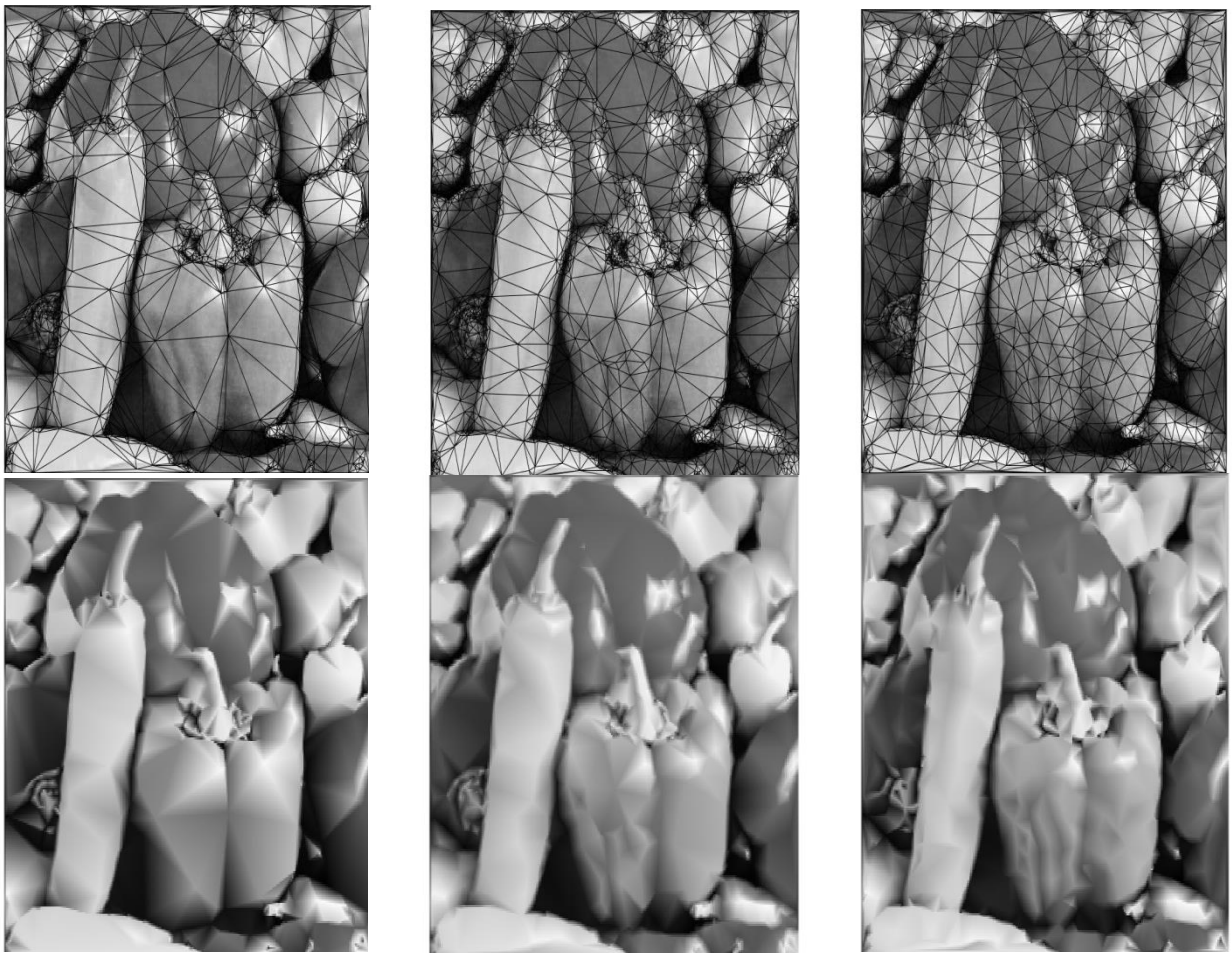
Figure5. Representation of image Peppers: (a) sample points; (b) recovery of image Peppers, PSNR =25.88

(a)MAE, PSNR = 24.96 (b) LMAE, PSNR = 24.05 (c) MAES, PSNR = 24.80

Figure 6. Triangulations using different refinement metrics for approximations of Lena image

Table 1: PSNR value comparison of image restoration by different methods.

sampling algorithm	Lena	Peppers
Q-Learning	24.71	25.88
MAE	24.96	24.01
LMAE	24.05	25.52
MAE	24.80	25.32



(a) MAE, PSNR = 24.01 (b) LMAE, PSNR = 25.52 (c) MAES, PSNR = 25.32

Figure 7. Triangulations using different refinement metrics for approximations of Peppers image

REFERENCES

[1] Voronin, V. V. Holographic representation in image processing tasks. Pattern Recognition and Image Analysis (Advances in Mathematical Theory and Applications) 11.1 (2001) 265-267.

[2] Ying, Liu, Surendra Ranganath, and Xiaofang Zhou. Wavelet-based image segment representation." Electronics Letters 38(19) (2002) 1091-1092.

[3] Monasse, Pascal, and Frederic Guichard. Fast computation of a contrast-invariant image representation. IEEE Transactions on Image Processing 9(5) (2000) 860-872.

[4] Gan, Tao, Yanmin He, and Weile Zhu. Fast $ M $-Term Pursuit for Sparse Image Representation. IEEE Signal Processing Letters 15 (2008) 116-119.

[5] He, Zhihai. Peak transform for efficient image representation and coding. IEEE Transactions on Image Processing 16(7) (2007) 1741-1754.

[6] Wallace, Gregory K. "Overview of the JPEG (ISO/CCITT) still image compression standard. Image Processing Algorithms and Techniques. 1244. International Society for Optics and Photonics, 1990.

[7] Flusser, Jan. Refined moment calculation using image block representation. IEEE Transactions on Image Processing 9(11) (2000) 1977-1978.

[8] Sullivan, Gary J., and Richard L. Baker. Efficient quadtree coding of images and video. IEEE Transactions on image processing 3(3) (1994) 327-331.

[9] Shukla, Rahul, et al. rate-distortion optimized tree-structured compression algorithms for piecewise polynomial images. IEEE transactions on image processing 14(3) (2005) 343-359.

[10] Scholefield, Adam, and Pier Luigi Dragotti. Quadtree structured image approximation for denoising and interpolation. IEEE transactions on image processing 23(3) (2014) 1226-1239.

[11] Burt, Peter J., and E. H. Edward. Adelson, The laplacian pyramid as a compact image code. IEEE Transactions on Communications 31(4) (1983) 532-540.

[12] Tran, Anh, et al. An efficient pyramid image coding system. ICASSP'87. IEEE International Conference on Acoustics, Speech, and Signal Processing. 12. IEEE, 1987.

[13] De Araújo, Bruno Rodrigues, et al. A survey on implicit surface polygonization. ACM Computing Surveys (CSUR) 47(4) (2015) 1-39.

[14] Bommes, David, et al. Quad-mesh generation and processing: A survey. Computer Graphics Forum. 32(6). (2013).

[15] Botsch, Mario, et al. Polygon mesh processing. CRC Press, 2010.

[16] Maglo, Adrien, et al. Progressive compression of manifold polygon meshes. Computers & Graphics 36(5) (2012) 349-359.

[17] Adams, Michael D. A flexible content-adaptive mesh-generation strategy for image representation. IEEE Transactions on Image Processing 20(9) (2011) 2414-2427.

[18] Terzopoulos, Demetri, and Manuela Vasilescu. Sampling and reconstruction with adaptive meshes. CVPR. 91 (1991).

[19] Courchesne, O., et al. Adaptive mesh generation of MRI images for 3D reconstruction of human trunk. International Conference Image Analysis and Recognition. Springer, Berlin, Heidelberg, 2007.
[20] Sarkis, Michel, and Klaus Diepold. Content adaptive mesh representation of images using binary space partitions. IEEE Transactions on Image Processing 18(5) (2009) 1069-1079.
[21] Joshi, Siddharth, and Stephen Boyd. Sensor selection via convex optimization. IEEE Transactions on Signal Processing 57(2) (2008) 451-462.
[22] Lawler, Eugene L., and David E. Wood. Branch-and-bound methods: A survey. Operations research 14(4) (1966) 699-719.
[23] Yao, Leehter, William A. Sethares, and Daniel C. Kammer. Sensor placement for on-orbit modal identification via a genetic algorithm. AIAA Journal 31(10) (1993) 1922-1928.
[24] Basirian, Saeed, and Alexander Jung. "Random walk sampling for big data over networks." 2017 International Conference on Sampling Theory and Applications (SampTA). IEEE, 2017.
[25] Abramenko, Oleksii, and Alexander Jung. Graph signal sampling via reinforcement learning. ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP). IEEE, 2019.
[26] Lu, Yongjie, et al. Energy-Efficient Depth-Based Opportunistic Routing with Q-Learning for Underwater Wireless Sensor Networks. Sensors 20(4) (2020) 1025.
[27] Xu, Qichao, Zhou Su, and Rongxing Lu. Game theory and reinforcement learning-based secure edge caching in mobile social networks. IEEE Transactions on Information Forensics and Security (2020).
[28] Liu, Libin, and Urbashi Mitra. On Sampled Reinforcement Learning in Wireless Networks: Exploitation of Policy Structures. IEEE Transactions on Communications 68(5) (2020): 2823-2837.
[29] Kaelbling, Leslie Pack, Michael L. Littman, and Andrew W. Moore. Reinforcement learning: A survey. Journal of artificial intelligence research 4 (1996) 237-285.
[30] Sutton, Richard S., and Andrew G. Barto. Reinforcement learning: An introduction. MIT Press, 2018.
[31] Watkins, Christopher JCH, and Peter Dayan. Q-learning. Machine learning 8(3-4) (1992) 279-292.
[32] Al, Walid Abdullah, and Il Dong Yun. Partial policy-based reinforcement learning for anatomical landmark localization in 3d medical images. IEEE Transactions on Medical Imaging 39(4) (2019) 1245-1255.
[33] Fisher, Ronald A. The use of multiple measurements in taxonomic problems. Annals of eugenics 7(2) (1936) 179-188.
[34] Weber, Allan G. The USC-SIPI image database version 5. USC-SIPI Report 315(1) (1997).
[35] da Silva, Eduardo Sant'Ana, Anderson Santos, and Helio Pedrini. Metrics For Image Surface Approximation Based On Triangular Meshes. Image Analysis & Stereology 37(1) (2018) 71-82.