# Active Perception in Computer Vision

Caus Danu

July 10, 2019

**Abstract**

Active Perception is a rapidly emerging field that deals with the dynamic actuation of sensors in order to acquire meaningful information from the environment. Active Vision strives to apply the principles of active perception in relation to computer vision. Currently, there are various state of the art solutions to the problem of active vision, ranging from POMDPs, with all their variations and approximation techniques, and specialized Neural Network based approaches. We argue that there is no best solution to the problem, each model having its own strengths and weaknesses and niche application targets.

## 1 Introduction

Till recently the field of perception was set up as a problem in which the agent tried to interpret in a static manner the sensor data coming from the environment. As practice shows, there are many situations however in which this is simply not effective, because of occlusions for instance and limited receptive field of the sensor itself. It is therefore much more desirable to be able to actively gather information from the environment via moving the sensor to different viewing poses. As a biological analogy, consider the human vision system i.e. HVS. Not only we as humans have two important and very complex sensors, namely our eyes, we also have the ability to move them via the so called saccades (see [1]). Through saccades we sample important information in order to understand the static and dynamic scene around us. No matter how sophisticated the human eye is, it is vital to physically move it around in order to guide attention and act in response to environmental stimuli. Hence, in the artificial realm of computer vision, researchers have created various methods and frameworks to emulate the same principle. The sensor, in our case a camera, is placed in various positions and orientations in space in order to minimize the perception uncertainty and create a model as accurate as possible of the true hidden state of the world. There are many different ideas to design the problem, ranging from applying Q-learning for energy efficient data collection (see [2]), to carefully and manually hand-crafting submodular sensor placement heuristic functions (see [3],

[4], [5]).The framework of choice in many cases is the ubiquitous Markov Decision Process, i.e. MDP. However, since we are dealing with only partial observations, uncertainties and beliefs, a POMDP or Partially Observable MDP is used most often (see [6], [7]). Since devices become increasingly affordable, with acceptable quality, they can be combined in systems and interact with one another while performing the task of perception. For such scenarios, a Decentralized POMDP is more suitable (theoretical background at [8]), that allows to reason about the overall joint belief state, using individual beliefs and actions of each entity in the system (see Lauri et al. [9], [10]). The problem with POMDP based frameworks is that complexity grows exponentially with the number of sensors and sensor poses (see [11], [12] regarding the PSPACE complexity of a POMDP and the curse of dimensionality). Even a relatively small state space becomes an issue for finding an exact solution (i.e. non-approximated solution) of the POMDP. Therefore, the problem becomes one of finding good approximations for specific scenarios and tasks, such that there are certain space and time guarantees of the final solution. In the next chapter, various such technics will be described in more detail.

# 2 Method description

This chapter aims to describe 3 main directions in active perception, out of the many available approaches and their numerous variations.

One of the important principles that researchers consider in the field of active vision is whether to reason about the task myopically or non-myopically. As the name implies, a myopic, short sighted approach is a greedy decision making procedure, using a next best view (NBV) strategy. A non-myopic method will make intermediary non-greedy decisions in order to increase the overall long term reward. In other words, a myopic system will care about short term rewards, whereas a non-myopic one will reason about a further horizon and long term reward maximization.

## 2.1 Nonmyopic View Planning

An example of non-myopic system is the one proposed by Atanasov et al. [13], where the task is to simultaneously detect and classify various objects along with their orientation. In other words, it is a dual problem of classification and pose estimation, which can potentially be very useful for robotics, for example in the grasping task of a robot manipulator.

The method can be concisely viewed in Figure 1. The general idea consists in training a data structure of relevant features, which the authors call a VP-Tree (i.e. viewpoint-pose tree). The VP-Tree provides a pose estimate in addition to detecting the object's class. The confidence of the result is encoded in the score that the VP-Tree assigns when the camera views the object

from a particular angle. The system than uses this score in order to create a plan of optimal points along a sphere surrounding the object of interest, which is placed at the center of the sphere. The algorithm continues to take measurements as long as the cost of making an incorrect decision/hypothesis $H(\hat{c}, \hat{r})$ is greater than the cost of one more measurement. In other words, the objective is to minimize the total cost:

$$E[J_M(\tau) + \lambda J_D(\hat{c}, \hat{r}, c, r)] \tag{1}$$

where $J_M$ is the cost of movement, $J_D$ is the cost of a decision, $\lambda$ is the relative importance weight of a correct decision versus cost of movement, c being the true class, r being the true orientation of the object and the $\hat{c}$ and $\hat{r}$ being the corresponding predicted/hypothesized terms.

One important thing to note is that the above function is not submodular as it is the case in many other approaches, that use for example: Mutual Information MI, or the Entropy function H(X). The submodularity property is important to quantify the utility of adding one more sensor (pose) and ensuring added value in terms of information gain (see [5] for more information on submodularity and adaptive submodularity).

There are some details that make this algorithm work faster than the usual theoretic expectation. First, the sphere along which the sensor moves is discretized in 42 possible view points. Secondly, the pose of the objects is also discretized in 6 possible cases: 0°, 60°, 120°, 180°, 240°, and 300°. Third, the tree data structure is trained offline. The last point in particular is very important, without which the utility score would be calculated very slowly. What is interesting is that, although the tree is trained offline using only simulated data, it performs well on real data as well, without the necessity to retrain or update the model. The experiments provide good results which can be seen in the Evaluation section below.

### 2.2 Learn-to-Score: Deep Learning approach

Unlike the previous approach, which requires approximately solving a POMDP over a number of state transitions, Hepp et al. [14] propose an alternative method using a trained Convolutional Neural Network (i.e. CNN). As expected, the heavy calculations are done during the training of the CNN, which allows the system to work very fast at test time, without any need for dynamic programming to calculate view utility. Moreover, this method is a greedy one, greedily selecting the next best view. Typically in the literature, an NBV method will be much like the one proposed by [13], just that it will pick actions greedily instead of looking at the long term reward. This CNN approach stands out because it works well for very big landscapes and offers promising results for unmanned air vehicle/UAV applications like drones, quadrocopters and the like. Figure 2 shows the structure of the CNN. The authors train the network using simulated data, just like [13] do and prove

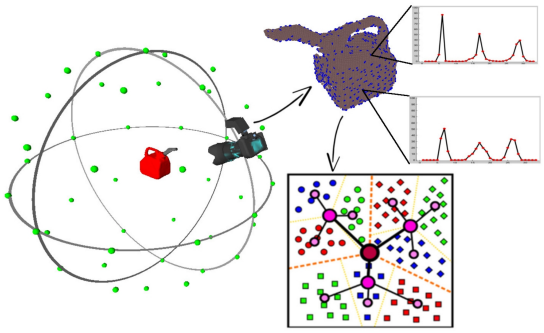

Figure 1: Figure taken from [13]. This figure illustrates the method of Atanasov et al. [13] which consists in allowing a camera to move on a sphere centered at the object location. The camera visits the desired viewpoints and obtains a corresponding point cloud for each one of them. Consequently, it ends up extracting relevant features as seen in the top right graph, which contribute to the construction of the VP-Tree in the bottom right corner.

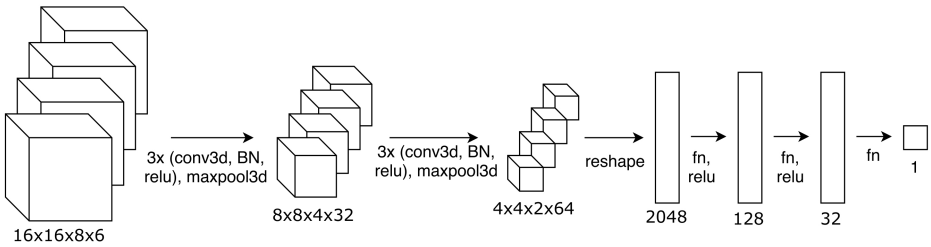


Figure 2: Figure taken from [14]. This figure illustrates the method of Hepp et al. [14] which consists in using a CNN designed to approximate an oracle utility function. This approach is substantially different than a dynamic programming way of solving an MDP, which is the usual way in which active vision problems are posed.

that the encoded knowledge is transferable to real world scenes as well. A very interesting detail is that the training is done on a very peculiar dataset named: "Washington2" (see [15]), but oddly enough, the network generalizes very well to all sorts of datasets, significantly different in their patterns of building height, distribution and geometry (ex: "SanFrancisco" dataset (from the broader [15] dataset) with landscape from San Francisco USA, which is notorious for being unlike any other city in the world).

The authors use an L2 loss function:

$$L(X, Y; \theta) = \sum_{n=1}^{N} \left\| f(X_i) - Y_i \right\|^2 + \lambda \|\theta\|^2 \tag{2}$$

where $\theta$ are the model parameters and $(X_i, Y_i)$ are the input and output

data points. More specifically: $X_i$ is the occupancy information and not the raw sensor data as one might initially assume, while $Y_i$ is the oracle's score as a numerical value.

### 2.3 Scheduling big sensor networks with real-time POMDPs

One more interesting example that shows how a POMDP, although exponential in nature, can be greatly simplified with the right assumptions and solved in real time for practical applications is the one from Vaisenberg et al. [16]. They have been able to actuate a big system of cameras for surveillance purposes, such that they acquire important information about salient persons and actions. This can be useful for security reasons in airports or any crowded areas. The cameras can pan, zoom and tilt and they have the purpose of taking high resolution images of peoples' faces who are of interest. Of course they can't just greedily zoom in at every face because then, the system won't extract useful information about moving entities and the statistics of their movement. In other more metaphorical words: the "dynamics" of the world would not be accurate, and this hinders future planning about when to pursue and when to refrain from taking greedy actions. Staying "UP", or zoomed out is also important even though it implies lower detail images, because it allows the algorithm to extract important semantics about the world itself and what might be of interest in the future. So the system tries to balance the need of some other applications of taking greedy actions and high detail images, and its own need of knowing the world so that it can act greedily at the right time. In a way, it is a flavour of the exploration-exploitation dilemma. Since the intention is to use many cameras, in the order of tens, possibly hundreds, and each camera has many regions in its field of view that it can zoom into, the POMDP would therefore be intractable. Hence, the authors took some precautionary, simplifying measures and assumptions: the cameras have non-overlapping views (see Figure 3) and the background algorithms extract semantics from the environment in the form of correlations (see Figure 4). The way the cameras are placed spatially is such that a moving entity captured by one camera would correlate with the same entity being captured by another camera at a later point in time. Other cameras however, probabilistically "know" in advance that they won't see the entity until some later point in time, because they might be uncorrelated with the camera that observes the entity currently. As a result, the system now can trim a lot of branches and nodes in the decision tree that are not of interest.

Having described the 3 methods above, the next chapter will show the results and performance they exhibit in simulation and in practice.

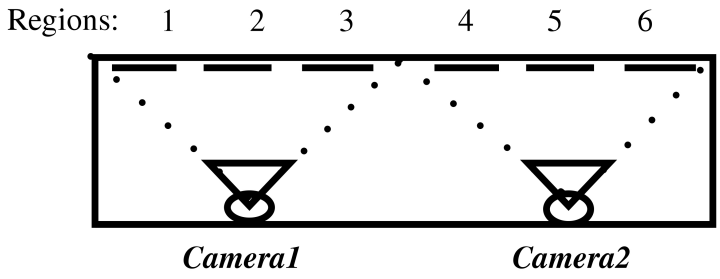


Figure 3: Figure taken from [16]. Long hallway continuously covered by cameras with non-overlapping fields of view as a simplifying assumption.

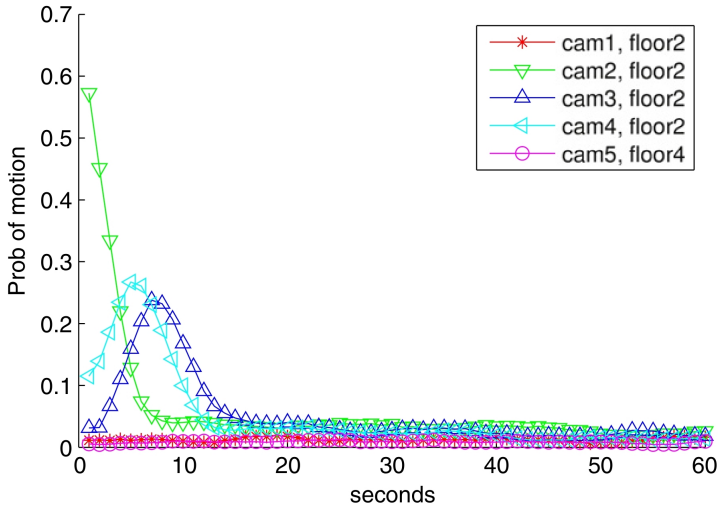


Figure 4: Figure taken from [16]. This figure illustrates the procedure of correlating camera data in order to predict what certain cameras will see in the future based on what other cameras observe currently. According to Vaisenberg et al. [16] this allows to significantly reduce the state space for the POMDP.

# 3 Experimental result and evaluation

## 3.1 Nonmyopic View Planning

For the object classification and pose estimation approach, the authors try to compare it with 3 other solutions:

- **Greedy Mutual Information**: greedily maximize the mutual information function.
- **Random Method**: random walk on the viewsphere while not revisiting the same viewpoints.

Table 1: Simulation **Accuracy** Results (created based on data from [13])

| Hypothesis | H(0°) | H(60°) | H(120°) | H(180°) | H(240°) | H(300°) | H(Other) |
|---|---|---|---|---|---|---|---|
| Static method | 60.35 | 53.90 | 51.49 | 49.13 | 56.11 | 54.29 | 89.87 |
| Random method | 73.78 | 70.34 | 70.75 | 66.97 | 68.76 | 71.85 | 92.33 |
| Greedy MI method | 82.63 | 80.14 | 76.93 | 75.60 | 75.29 | 81.78 | 94.65 |
| **NVP method** | 87.98 | 83.78 | 82.81 | 82.61 | 78.73 | 81.60 | 93.20 |

Table 2: Real-World **Accuracy** Results (created based on data from [13])

| Hypothesis | H(0°) | H(60°) | H(120°) | H(180°) | H(240°) | H(300°) | H(Other) |
|---|---|---|---|---|---|---|---|
| **NVP** Accuracy | 87.5 | 80.0 | 72.5 | 70.0 | 75.0 | 72.5 | 98.05 |

Table 3: Table refactored from [14]. Efficiency Metric of the CNN utility function on the *3D Street View* dataset [15], which contains models of different cities. As it can be observed, although the other functions are hand-crafted, trying to approximate a ground truth oracle is more effective.

| Efficiency Metric | Washington2 | Washington1 | Paris | SanFrancisco | Neighborhood |
|---|---|---|---|---|---|
| Frontier | 0.40 | 0.29 | 0.57 | 0.09 | 0.27 |
| AverageEntropy [17] | 0.26 | 0.36 | 0.32 | 0.30 | 0.50 |
| ProximityCount [17] | 0.52 | 0.47 | 0.37 | 0.23 | 0.60 |
| **CNN [14]** | **0.91** | **0.88** | **0.87** | **0.77** | **0.74** |
| Oracle (GT access) | 1.00 | 1.00 | 1.00 | 1.00 | 1.00 |

- **Static Method**: take a single measurement at the initial position and decide based on that greedily.

As Table 1 shows, the classification is done correctly and with high confidence as well. However, note that the results are not much higher than the well established Greedy Mutual Information approach. The authors acknowledge this and say that the intention of their method is to have a more robust stopping criterion, i.e. adaptively decide when it makes sense to stop taking measurements based on the observations received online.

Since all the main experiments and training were done in simulations, the authors also show that the results are transferable to real world practical applications using a PR2 robot (see Table 2).

### 3.2 Learn-to-Score: Deep Learning approach

Regarding the CNN attempt to approximate a ground truth oracle function, Table 3 shows that it indeed manages to outperform significantly some of the most prominent hand-crafted solutions out there. Not only the score is accurate, but the decision time is better as Table 4 points out. These results one more time convince us of the efficiency of neural nets over vanilla methods in computer vision. However, there are certain trade-offs that will be discussed in the future section.

Table 4: Table refactored from [14]. One of the advantages of not using an online dynamic programming approach is the lightning fast response time, in this case equal to the inference time of the neural network.

| | Frontier | ProximityCount | AverageEntropy | CNN [14] |
|---|---|---|---|---|
| Time in s | 0.61 | 5.89 | 8.35 | **0.57** |

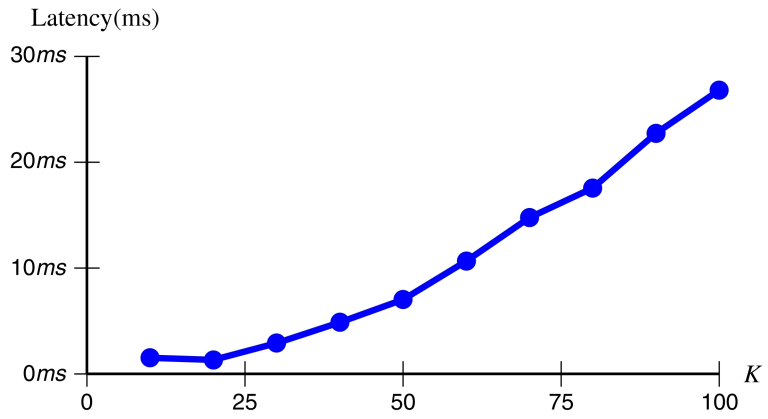


Figure 5: Figure taken from [16]. Using global semantics and local lookup tables, Vaisenberg et al. [16] have been able to achieve real-time performance using a very impressive number of cameras $\boldsymbol{K}$. As it can be seen, the marginal cost of adding one more camera is very small and does not increase the latency beyond what can be considered as real-time.

### 3.3 Scheduling big sensor networks with real-time POMDPs

Continuing with the last surveillance system in the above chapter, it can be observed in Figure 5 that it indeed performs very well in terms of latency, with a very big number of cameras K. For a non-approximated solution, even a size of 10 to 30 would already be a big problem for real time operation. Figure 6 illustrates that this is indeed the case, and note that the exact solution there, using an exhaustive tree search, only considers a depth of 2 seconds look-ahead to calculate the utility score. If a bigger depth of the tree would be considered, the latency would have a huge increase.

Considering these results, we will discuss the advantages and disadvantages in the following chapter and reason about when might one consider a certain approach over another.

## 4 Discussion

In this section we will discuss the strengths and weaknesses of the already presented methods.

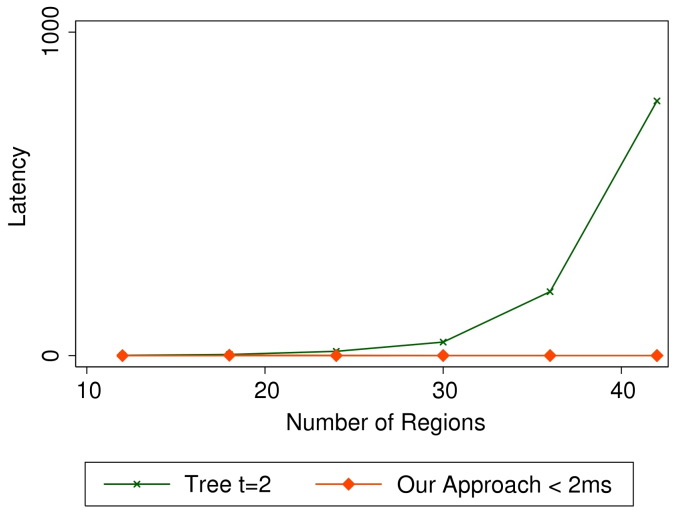


Figure 6: Figure taken from [16]. This figure motivates the need for approximating the solution of the POMDP using the method of Vaisenberg et al. [16]. Otherwise, the exact solution requires exponential computation time with each new added camera/region to monitor, and the solution cannot be considered real-time anymore.

The method of Atanasov et al. [13] has the advantage that it tackles two aspects of computer vision which are very useful for practical applications: detection and pose estimation. Bridging the gap between these two problems would significantly contribute to solving the grasping problem in robotics. The disadvantage is that, although the results are promising and resemble the best ones from other methods, the authors did not factor in the problem of segmentation. All simulations are carefully constructed with objects on a table, in such a way that they do not require complex segmentation. It was not the scope of the work to create a customized segmentation component, but one can argue that it is a good way to reduce the complexity of the POMDP and obtain a more complete and practical system.

The CNN approach on the other hand, is very oriented towards solving practical issues. The authors argue that all POMDP approaches can only work for small scale problems and therefore, in order to be able to explore huge landscape, one needs a new way to tackle the problem, which is a trained neural net. What can be mentioned is that the CNN will try to approximate an oracle function, which knows the true utility score of a view, and the resulting function will presumably not have the exact theoretical guarantees as a handcrafted utility function might have (or as the true oracle has). However, as it was seen in the evaluation section, in practice, even without those theoretical proofs, the neural net performs better than well established methods and heuristic functions. A potential drawback is that there can

always be some edge case that is not sufficiently addressed and different training experiments of the CNN would result in different behaviours, since the training process is stochastic by nature. A hand-crafted function is by definition more stable in its behaviour, at least until scientists will prove more things about neural nets and shed more light on the "black box" aspect that they are stigmatized with.

The third system, aiming to create a framework backbone on top of which many applications can run, tries to combine the best of both worlds:

- Make a system that can scale
- Still use a POMDP to control the sensors

A huge advantage is that their sensor-correlation model is real time for large state spaces. The authors manage to find important simplifying aspects of the problem. However, this means that the model will only work for specialized environments and tasks, in this case: a people surveillance task. Many other problems however might not fulfil the underlying prerequisites to be modelled in a similar fashion. And since people surveillance might trigger ethical concerns, the reader will be positively surprised and rest-assured that the same correlation model can be applied to traffic lights control for the purpose of reducing traffic jam amongst other things.

# 5 Conclusion

In conclusion, what can be said about active perception is that it is a field with a lot of potential applications, that is currently receiving a lot of attention from the research community. Up until recently it was considered a niche topic within the computer vision realm, but things have changed with the advent of cheaper, high-quality sensing devices. The POMDP approach, since it is considered a PSPACE-complete problem (see [11] and [18]), means that approximated solutions will be developed for concrete tasks of active perception, on a case by case basis. In other words, there will not be a one fits all solution, till the time when POMDPs are theoretically solved, if this ever happens. Other solutions like neural nets and various efficient data structures and clever algorithmic tricks will always remain an option and that is what makes the field so reach, interesting, diverse and creative from an engineering point of view. We convinced ourselves of the richness of applications and diversity of solutions throughout the chapters. There is more work to be done however, especially in the decentralized POMDP domain, where things are still in their infancy in both theoretical as well as practical terms.

## References


[1] Martin Rolfs. Attention in active vision: A perspective on perceptual continuity across saccades. *Perception*, 44(8-9):900–919, 2015.

[2] Mario Di Francesco, Kunal Shah, Mohan Kumar, and Giuseppe Anastasi. An adaptive strategy for energy-efficient data collection in sparse wireless sensor networks. In *European Conference on Wireless Sensor Networks*, pages 322–337. Springer, 2010.

[3] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical programming*, 14(1):265–294, 1978.

[4] Uriel Feige. A threshold of ln n for approximating set cover. *Journal of the ACM (JACM)*, 45(4):634–652, 1998.

[5] Daniel Golovin and Andreas Krause. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research*, 42:427–486, 2011.

[6] Robert Eidenberger and Josef Scharinger. Active perception and scene modeling by planning with probabilistic 6d object poses. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1036–1043. IEEE, 2010.

[7] Mohan Sridharan, Jeremy Wyatt, and Richard Dearden. Planning to see: A hierarchical approach to planning visual actions on a robot using POMDPs. *Artificial Intelligence*, 174(11):704–725, 2010.

[8] Frans A Oliehoek, Christopher Amato, et al. *A concise introduction to decentralized POMDPs*, volume 1. Springer, 2016.

[9] Mikko Lauri, Eero Heinänen, and Simone Frintrop. Multi-robot active information gathering with periodic communication. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 851–856. IEEE, 2017.

[10] Mikko Lauri, Joni Pajarinen, and Jan Peters. Information gathering in decentralized POMDPs by policy graph improvement. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pages 1143–1151. International Foundation for Autonomous Agents and Multiagent Systems, 2019.

[11] Christos H Papadimitriou and John N Tsitsiklis. The complexity of markov decision processes. *Mathematics of operations research*, 12(3):441–450, 1987.

[12] Guy Shani, Ronen I Brafman, and Solomon E Shimony. Prioritizing point-based POMDP solvers. In *European Conference on Machine Learning*, pages 389–400. Springer, 2006.

[13] Nikolay Atanasov, Bharath Sankaran, Jerome Le Ny, George J Pappas, and Kostas Daniilidis. Nonmyopic view planning for active object classification and pose estimation. *IEEE Transactions on Robotics*, 30(5):1078–1090, 2014.

[14] Benjamin Hepp, Debadeepta Dey, Sudipta N Sinha, Ashish Kapoor, Neel Joshi, and Otmar Hilliges. Learn-to-score: Efficient 3d scene exploration by predicting view utility. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 437–452, 2018.

[15] Amir R Zamir, Tilman Wekel, Pulkit Agrawal, Colin Wei, Jitendra Malik, and Silvio Savarese. Generic 3d representation via pose estimation and matching. In *European Conference on Computer Vision*, pages 535–553. Springer, 2016.

[16] Ronen Vaisenberg, Alessio Della Motta, Sharad Mehrotra, and Deva Ramanan. Scheduling sensors for monitoring sentient spaces using an approximate POMDP policy. *Pervasive and Mobile Computing*, 10:83–103, 2014.

[17] Stefan Isler, Reza Sabzevari, Jeffrey Delmerico, and Davide Scaramuzza. An information gain formulation for active volumetric 3d reconstruction. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3477–3484. IEEE, 2016.

[18] Sanjeev Arora and Boaz Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009.