Matrix Completion as a Post-Processing Technique for Probabilistic Roadmaps

Joel M. Esposito[1] and John N. Wright[2]

[1] United States Naval Academy, Annapolis, MD, USA
esposito@usna.edu,
[2] Columbia University, New York, NY, USA
jw2966@columbia.edu

Abstract. This paper describes a novel post-processing algorithm for probabilistic roadmaps (PRMs), inspired by the recent literature on matrix completion. We argue that the adjacency matrix associated with real roadmaps can be decomposed into the sum of low-rank and sparse matrices. Our method, based on Robust Principal Component Analysis (RPCA), numerically computes a relaxation of this decomposition by solving a convex optimization problem–even when most of the entries in the adjacency matrix are unknown. Given a PRM with n vertices and $O(n \log^2 n)$ collision-checked candidate edges, the algorithm is able to estimate the status of all $n(n-1)/2$ possible edges in the full road map with high accuracy, without performing any additional collision checks. Numerical experiments on problems from the Open Motion Planning Library indicate that they posses the requisite low-rank plus sparse structure; and that after checking 5% of the possible edges, the algorithm estimates the full visibility graph with 96% accuracy. We point to results from the graph clustering literature that relate the required number of edge checks to various graph parameters, and relate them to motion planning geometry. Based on numerical experiments, we propose sharper estimates of the error as a function of number of edge checks than what is previously reported. The practical utility of the algorithm is that average path length across the resulting denser edge set is significantly shorter (at the cost of somewhat increased spatial complexity and query times). An ancillary benefit is that the resulting low-rank plus sparse decomposition readily reveals information about that would be otherwise difficult to compute, such as the number of convex cells in free configuration space and the number of vertices in each. Perhaps more importantly, however, we believe that this novel connection between motion planning and matrix completion–two previously disparate lines of research–provides a new perspective on sampling-based motion planning and may serve to guide future algorithm development.

1 Introduction

Sampling-based motion planning techniques, such as the *probabilistic road map* (PRM) method [1], efficiently generate a graph, G_{prm}, which approximates the

free configuration space by randomly sampling a discrete set of free configurations (vertices) and collision checking only a small sub-set of the possible edges between them. This paper describes a novel post-processing step for PRMs, inspired by the success of so-called matrix completion techniques [2] which recover large, structured, matrices from a small set of randomly observed entries. Specifically, given a PRM with n vertices and $O(n \log^2 n)$ collision-checked candidate edges, we are able to estimate, with high probability, the status of nearly all of the remaining $n(n-1)/2$ possible edges in the full visibility graph without additional collision checks. From a computational perspective, the method involves solving a convex optimization problem, whose unique optimum can be found efficiently, without the need for parameter tuning or *a priori* knowledge of the solution. The practical benefit of post-processed denser edge set is that it contains shorter paths–at the cost of somewhat increased spatial complexity and query times.

At a more fundamental level we believe that the primary contribution of this work is the previously unexploited insight that the adjacency matrix can be decomposed into sparse and low-rank terms, which encode information about problem difficulty. The empirical successes of PRM and other sampling-based planners, such as RRT [3] or EST [4], at estimating the connectivity of realistic environments, using very few collision checks, has outstripped our theoretical understanding. Several analyses of PRM completeness [5, 6], convergence [7, 8], and optimality [9] have been offered. Yet most require knowing the geometric parameters of a particular solution path, such as obstacle clearance [9] or path length [7]. In [10] it was hypothesized that the apparent success of PRM can be attributed to the favorable properties possessed by most real-world environments, called expansiveness. While this notion does not require knowledge of a solution path, its parameters are difficult to compute for high dimensional environments. However, some easily computable properties of our low-rank plus sparse decomposition provide a similar characterization of problem difficulty.

Moreover, we hope this novel application of matrix completion to the motion planning problem will inspire cross-pollination between these previously disparate fields and provide new perspective on sampling-based motion planning.

The remainder of this paper is organized as follows. Section 2.1 reviews results on matrix completion with a focus on robust principal component analysis; Section 2.2 reviews the PRM algorithm; and Section 2.3 discusses connections between these concepts. Section 3 and 4 describe our post-processing technique along with some performance analysis in Sect. 5. Section 6 presents the results of our computational experiments on a set of benchmark problems. Finally, Section 7 places our results in context and presents a program for future work.

2 Background

2.1 Matrix recovery via convex optimization

In many areas of engineering, we are confronted with problems of missing data. Examples include predicting how users will rate novel items on a website, determining the relative position of sensors in a large network, and filling in missing values in a degraded digital image. We can formalize these problems as follows: in each, we observe a small subset $\Omega \subseteq [n] \times [n]$ of the entries of a large matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$, and the goal is to fill in the remaining entries of $\boldsymbol{A}$. Clearly, without prior knowledge about the structure of $\boldsymbol{A}$, this problem is ill-posed.

Fortunately, in many applications, the target matrix $\boldsymbol{A}$ is known to be approximately *low-rank*. In this situation, the problem becomes well-posed: provided $\boldsymbol{A}$ is not too concentrated on any row or column[3], only about $nr \log^2 n$ observations are needed to uniquely determine the matrix $\boldsymbol{A}$. This is nearly the same order as the number of degrees of freedom in a rank-r matrix– $O(nr)$. Moreover, not only is the problem well-posed but seminal results in theory of *matrix completion* show that it can be solved using efficient, well-structured algorithms based on convex optimization [2, 11–14].

Many problems in imaging and statistics pose an additional challenge: gross errors due to sensor noise, partial object occlusions, etc. This situation can be modeled as follows. Instead of assuming that $\boldsymbol{A}$ is low-rank, we assume that $\boldsymbol{A}$ is a *superposition* of a low-rank matrix $\boldsymbol{L}$ and a *sparse* error matrix $\boldsymbol{S}$: $\boldsymbol{A} = \boldsymbol{L} + \boldsymbol{S}$.

Unfortunately, the rank and sparsity (as measured by the vector ℓ^0 norm) are not convex functions. Instead, ℓ^1 is known to be the tightest convex relation of the sparsity criteria; and the *nuclear norm* $\|\boldsymbol{L}\|_* = \sum_i \sigma_i(\boldsymbol{L})$ –the sum of the singular values–is used as a convex relaxation of the rank function. Both $\boldsymbol{L}$ and $\boldsymbol{S}$ can be recovered by solving a convex optimization problem [15, 16]:

$$\min_{\boldsymbol{L},\boldsymbol{S}} \|\boldsymbol{L}\|_* + \lambda \|\boldsymbol{S}\|_1 \quad \text{such that} \quad P_{\Omega_{obs}}(\boldsymbol{S} + \boldsymbol{L}) = P_{\Omega_{obs}}(\boldsymbol{A}). \tag{1}$$

The parameter λ balances between rank and sparsity. Fortunately, theory provides good guidance for choosing λ: $O(1/\sqrt{n})$ is shown to work under broad conditions in [16]. $P_{\Omega_{obs}}$ is the orthoprojector onto the set of observed entries:

$$[P_{\Omega_{obs}}(\boldsymbol{A})]_{ij} = \begin{cases} \boldsymbol{A}_{ij} & (i,j) \in \Omega_{obs} \\ 0 & \text{else} \end{cases} \tag{2}$$

In words, this constraint forces $\boldsymbol{L} + \boldsymbol{S}$ to agree with the observed entries in $\boldsymbol{A}$. The past ten years have seen a tremendous development of the theory [14–18], algorithms [19, 20], as well as applications for robust matrix recovery–see [21] for an overview.

[3] This condition is needed to rule out very sparse $\boldsymbol{A}$. For example, if $\boldsymbol{A}$ has only one nonzero entry, we would need to sample exhaustively to locate it. When $\boldsymbol{A}$ is not too concentrated on any row or column, every entry carries information about the global structure of $\boldsymbol{A}$, and only a few entries are needed to correctly complete it.

2.2 Motion planning with probabilistic roadmaps

Some of the earliest work on motion planning [22] introduced the idea of a *roadmap*, which abstracts the robot's free configuration space, C_{free}, as a graph, G, whose vertices $V = \{v_1, \ldots, v_n\}$ are configurations in C_{free} and whose edges $E = \{e_{ij}\}$ are simple paths connecting vertices. Good roadmaps satisfy two properties [23]: *accessibility* means any point in C_{free} can be easily connected to a vertex; and *connectivity* means the graph has the same number of connected component as the free configuration space. Unfortunately computing an exact solution to the motion planning problem is known to be PSPACE-hard [24] and the complexity of complete algorithms, such as [25], makes them impractical. Sampling-based motion planning methods, such as PRM [1], trade a reduction in computation time for weaker (probabilistic) completeness. A generalization of the PRM algorithm is outlined in Fig. 1, and a sample PRM is shown Fig. 2 (left).

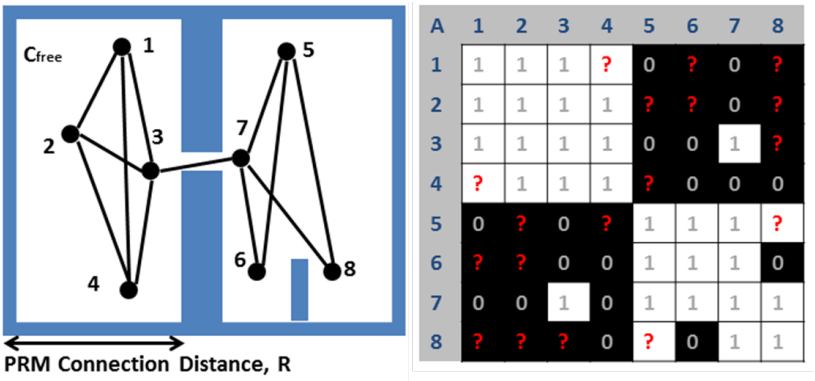
We refer to several graphs throughout this paper, all of which share the same randomly generated vertex set V but differ in their edges sets. G_{full}, termed the full visibility graph on V, is the graph created by the algorithm in Fig. 1 when the function *CandidateVertices* returns all other vertices. This is identical to the simplified-PRM algorithm [7], which results in $O(n^2)$ calls to *CollisionFree*– known to be the most computationally expensive step [9].

G_{prm} is the graph returned by other variants of PRM, when more restrictive heuristics are used for *CandidateVertices*. For example, the range-limited variant [26] only checks vertices in a ball of (possibly varying) radius R about v_i; another uses the K-nearest vertices [26]. The motivation is to limit the number of calls to *CollisionFree* to $O(n \log n)$ [9]. However, in doing so they create a sparse graph whose edges are a subset of G_{full}; and therefore G_{prm} has at least as many connected components and the length of the resulting paths are at least as large G_{full}. In fact, under many heuristics, the resulting path lengths are not even asymptotically optimal [9].

Generic Probabilistic Road Map (g-PRM): Preprocessing Phase

GenerateVertices $V = \{v_1, \ldots, v_n\} \in C_{free}$
Initialize edges $E = \emptyset$
for $i = 1, \ldots, n$ **do**
  $U \leftarrow$ *CandidateVertices*(v_i)
  for $\forall v_j \in U$ **do**
    if *CollisionFree* (v_i, v_j) **then**
      add edge e_{ij} to E
    end if
  end for
end for
RETURN $G = \{V, E\}$

Fig. 1. A generalization of the PRM pre-processing algorithm.



A	1	2	3	4	5	6	7	8
1	1	1	1	?	0	?	0	?
2	1	1	1	1	?	?	0	?
3	1	1	1	1	0	0	1	?
4	?	1	1	1	?	0	0	0
5	0	?	0	?	1	1	1	?
6	?	?	0	0	1	1	1	0
7	0	0	1	0	1	1	1	1
8	?	?	?	0	?	0	1	1

Fig. 2. The graph generated by simplified-PRM for a point robot in a simple 2-D environment (left) and its adjacency matrix (right). Note how the adjacency matrix is the superposition of a rank-2 block diagonal matrix and a sparse matrix with non-zero entries at (7,3), (3,7), (6,8) and (8,6). The question marks represent edges that the range-limited variant of PRM would not collision check.

2.3 Connections: matrix completion and PRM

Our goal is to compute a graph $\hat{G}$ that estimates the edge structure in G_{full} based on the limited edge information in G_{prm}, using the low-rank plus sparse matrix completion technique outlined in Sect. 2.1. One way to represent a graph is as an $n \times n$ adjacency matrix, $\boldsymbol{A}$, where $A_{ij} = 1$ if $e_{ij} \in E$, and $A_{ij} = 0$ otherwise. Figure 2 (left) depicts G_{full}; in the right panel, the shading indicates the corresponding adjacency matrix, $\boldsymbol{A}_{full}$, with black cells indicating 0 entries and white cells indicating 1 entries. The entries marked "?" will be discussed later.

We argue that in general $\boldsymbol{A}_{full}$ can be expressed as the sum of low rank and sparse matrices. To see this, consider Figure 2. Within each convex cell of C_{free} the vertices form a connected cluster. Ignoring entries $(3,7)$ and $(6,8)$, for a moment, $\boldsymbol{A}_{full}$ has an underlying block diagonal structure shaded in white (under a particular choice of vertex labeling), whose rank is equal to the number of convex cells of C_{free} that contain vertices. Note that this low-rank property requires adopting the convention $A_{ii} = 1$. The sparse matrix can be used to model the connecting edges *between* the convex cells, such as entry $(3,7)$, as well as to capture the effect of small occlusions, as in the case of $(6,8)$. Loosely, the accessibility property of roadmaps results in a low-rank structure, while the connectivity property necessitates the sparse component.

Now consider a G_{prm} produced when $CandidateVertices(q_i) = \{q_j | \, \|q_i - q_j\| \leq R\}$ (where R is shown in Fig. 2-left). The matrix entries of A_{prm} in the lower panel marked with "?" are unknown, e.g. the distance between vertex 1 and 4 is greater than R so *CollisionFree* will not be called for this pair. G_{prm} still

satisfies the accessibility and connectivity criteria yet the resulting paths would be longer in some cases. For example, since $e_{1,4}$ is not included in the sub-graph, a query to connect 1 to 4 would be routed via 2.

The RPCA method in Sect. 2.1 can be used to compute an estimate, $\hat{\boldsymbol{A}}$, of $\boldsymbol{A}_{full}$ using a limited set of observed entries such as those in $\boldsymbol{A}_{prm}$. Of course there are two notable differences between matrix completion and PRM. First is the way in which negative answers to *CollisionFree* are used. In most PRM algorithms if *CollisionFree*(q_i, q_j) is false, that information is essentially discarded; yet for matrix completion, this observation represents an additional constraint for the optimization problem. Second, proofs of the completeness of RPCA assume that the set of observations, Ω_{obs}, is selected at random, uniformly over $[n] \times [n]$ rather than using the geometric heuristic employed in PRM.

3 Algorithm overview

Given a motion planning problem, we run g-PRM where *CandidateVertices* will simply be $F \cdot n(n-1)/2$ vertex pairs selected uniformly at random. $F \in (0, 1]$ is called the *sample fraction*. Using the edge information observed during PRM's *CollisionFree* step, including negative results, we assemble the constraint (1). Since the graph is undirected, we add the corresponding symmetric entries of $\boldsymbol{A}$ to the set of observations Ω_{obs}. We then use the numerical implementation described in Sect. 4 to solve the optimization problem (1) with two additional constraints:

$$L_{ii} = 1 \qquad \forall\, i \in [n], \tag{3}$$

$$-1 \leq S_{ij} \leq 1 \qquad \forall\, i, j \in [n]. \tag{4}$$

We round the final estimate $\hat{\boldsymbol{A}} = round(\boldsymbol{L} + \boldsymbol{S})$ to ensure it is a binary matrix.

The additional constraints are motivated by the fact that the objective function in (1) is merely a convex relaxation of the low-rank plus sparse model. Constraint (3) allows on-block observed "1"s to be included in $\boldsymbol{L}$ for "free" rather than pay a penalty for including them in $\boldsymbol{S}$, since the nuclear norm of a block diagonal matrix is the same as that of the identity matrix (n). It also incentivizes the algorithm to extrapolate by completing the blocks when possible. It is worth noting that while a true rank minimizing $\boldsymbol{L}$ would have as few blocks as possible, the nuclear norm of a binary block diagonal matrix is n regardless of the number of blocks it contains so there is less incentive to "over generalize" by, for example, setting $\boldsymbol{L}$ to the all ones matrix. Finally, combined with the objective function, it effectively constrains $0 \leq L_{ij} \leq 1$. Constraint (4) is simply a convex relaxation of the property that the entries of S should be strictly $\{-1, 0, 1\}1$.

4 Numerical implementation

We developed a numerical solver for (1) based on the Alternating Directions Method of Multipliers (ADMM) [27, 28]. ADMM is a flexible framework for

constrained optimization, which is especially effective for problems in which the objective is separable–it can be decomposed into a sum of functions on blocks of variables [29]. ADMM has been successfully applied for RPCA and related problems in [19, 20]. A virtue of ADMM in this setting is that it allows us to easily handle constraints such as (3) and (4). This is accomplished by *splitting*: we introduce auxiliary variables $\bar{\boldsymbol{L}}, \tilde{\boldsymbol{L}}, \bar{\boldsymbol{S}}, \tilde{\boldsymbol{S}}$, and solve the equivalent problem

$$\begin{aligned} \min \|\boldsymbol{L}\|_* + \lambda \|\boldsymbol{S}\|_1 \quad \text{s.t.} \quad & \boldsymbol{L} = \bar{\boldsymbol{L}},\ \boldsymbol{S} = \bar{\boldsymbol{S}},\ \mathcal{P}_\Omega[\bar{\boldsymbol{L}} + \bar{\boldsymbol{S}}] = \boldsymbol{A}, \\ & \tilde{\boldsymbol{L}} = \bar{\boldsymbol{L}},\ \tilde{\boldsymbol{S}} = \bar{\boldsymbol{S}},\ \tilde{\boldsymbol{L}} \in \mathfrak{L},\ \tilde{\boldsymbol{S}} \in \mathfrak{S} \end{aligned} \tag{5}$$

Here, $\mathfrak{L}$ and $\mathfrak{S}$ are convex sets which encode any additional constraints on $\boldsymbol{L}$ and $\boldsymbol{S}$, such as the constraint that the diagonal elements of $\boldsymbol{L}$ equal one, interval constraints, etc. While splitting increases the number of variables, it makes it much easier to develop efficient numerical methods, by grouping the "difficult" parts of the objective and the constraint in a way that allows them to be handled independently. To accomplish this, we introduce Lagrange multipliers $\boldsymbol{\Lambda}$, $\boldsymbol{\Gamma}_L$, $\boldsymbol{\Gamma}_S$, $\boldsymbol{\zeta}_L$, $\boldsymbol{\zeta}_S$ corresponding to the equality constraints in (5), and write the *Augmented Lagrangan* for this modified problem as

$$\begin{aligned} &\mathcal{L}_\nu(\boldsymbol{L}, \boldsymbol{S}, \bar{\boldsymbol{L}}, \bar{\boldsymbol{S}}, \boldsymbol{\Lambda}, \boldsymbol{\Gamma}_L, \boldsymbol{\Gamma}_S, \boldsymbol{\zeta}_L, \boldsymbol{\zeta}_S) \\ &= \|\boldsymbol{L}\|_* + \lambda \|\boldsymbol{S}\|_1 + \left\langle \mathcal{P}_\Omega[\bar{\boldsymbol{L}} + \bar{\boldsymbol{S}}] - \boldsymbol{A}, \boldsymbol{\Lambda} \right\rangle + \left\langle \boldsymbol{L} - \bar{\boldsymbol{L}}, \boldsymbol{\Gamma}_L \right\rangle + \left\langle \boldsymbol{S} - \bar{\boldsymbol{S}}, \boldsymbol{\Gamma}_S \right\rangle \\ &\quad + \tfrac{\nu}{2} \left\| \boldsymbol{L} - \bar{\boldsymbol{L}} \right\|_F^2 + \tfrac{\nu}{2} \left\| \boldsymbol{S} - \bar{\boldsymbol{S}} \right\|_F^2 + \tfrac{\nu}{2} \left\| \tilde{\boldsymbol{L}} - \bar{\boldsymbol{L}} \right\|_F^2 + \tfrac{\nu}{2} \left\| \tilde{\boldsymbol{S}} - \bar{\boldsymbol{S}} \right\|_F^2 \\ &\quad + \tfrac{\nu}{2} \left\| \mathcal{P}_\Omega[\bar{\boldsymbol{L}} + \bar{\boldsymbol{S}}] - \boldsymbol{A} \right\|_F^2 . \end{aligned} \tag{6}$$

The basic iteration in ADMM then consists in minimizing $\mathcal{L}$ with respect to each block $(\boldsymbol{L}, \boldsymbol{S}, \tilde{\boldsymbol{L}}, \tilde{\boldsymbol{S}})$ and $(\bar{\boldsymbol{L}}, \bar{\boldsymbol{S}})$ of primal variables individually, and then updating the Lagrange multipliers $(\boldsymbol{\Lambda}, \boldsymbol{\Gamma}_L, \boldsymbol{\Gamma}_S)$ by one step of gradient ascent, with a very particular choice of the step size:

$$\begin{aligned} &(\boldsymbol{L}^{(k+1)}, \boldsymbol{S}^{(k+1)}, \tilde{\boldsymbol{L}}^{(k+1)}, \tilde{\boldsymbol{S}}^{(k+1)}) \\ &= \arg\min_{\boldsymbol{L}, \boldsymbol{S}} \mathcal{L}_\nu \left(\boldsymbol{L}, \boldsymbol{S}, \tilde{\boldsymbol{L}}, \tilde{\boldsymbol{S}}, \bar{\boldsymbol{L}}^{(k)}, \bar{\boldsymbol{S}}^{(k)}, \boldsymbol{\Lambda}^{(k)}, \boldsymbol{\Gamma}_L^{(k)}, \boldsymbol{\Gamma}_S^{(k)}, \boldsymbol{\zeta}_L^{(k)}, \boldsymbol{\zeta}_S^{(k)} \right), \end{aligned} \tag{7}$$

$$\begin{aligned} &(\bar{\boldsymbol{L}}^{(k+1)}, \bar{\boldsymbol{S}}^{(k+1)}) \\ &= \arg\min_{\bar{\boldsymbol{L}}, \bar{\boldsymbol{S}}} \mathcal{L}_\nu \left(\boldsymbol{L}^{(k+1)}, \boldsymbol{S}^{(k+1)}, \tilde{\boldsymbol{L}}^{(k+1)}, \tilde{\boldsymbol{S}}^{(k+1)}, \bar{\boldsymbol{L}}, \bar{\boldsymbol{S}}, \boldsymbol{\Lambda}^{(k)}, \boldsymbol{\Gamma}_L^{(k)}, \boldsymbol{\Gamma}_S^{(k)}, \boldsymbol{\zeta}_L^{(\boldsymbol{k})}, \boldsymbol{\zeta}_S^{(k)} \right) \\ &\boldsymbol{\Lambda}^{(k+1)} = \boldsymbol{\Lambda}^{(k)} + \nu \left(\mathcal{P}_\Omega \left[\bar{\boldsymbol{L}}^{(k+1)} + \bar{\boldsymbol{S}}^{(k+1)} \right] - \boldsymbol{A} \right) \\ &\boldsymbol{\Gamma}_L^{(k+1)} = \boldsymbol{\Gamma}_L^{(k)} + \nu \left(\boldsymbol{L}^{(k+1)} - \bar{\boldsymbol{L}}^{(k+1)} \right), \quad \boldsymbol{\zeta}_L^{(k+1)} = \boldsymbol{\zeta}_L^{(k)} + \nu \left(\tilde{\boldsymbol{L}}^{(k+1)} - \bar{\boldsymbol{L}}^{(k+1)} \right) \\ &\boldsymbol{\Gamma}_S^{(k+1)} = \boldsymbol{\Gamma}_S^{(k)} + \nu \left(\boldsymbol{S}^{(k+1)} - \bar{\boldsymbol{S}}^{(k+1)} \right), \quad \boldsymbol{\zeta}_S^{(k+1)} = \boldsymbol{\zeta}_S^{(k)} + \nu \left(\tilde{\boldsymbol{S}}^{(k+1)} - \bar{\boldsymbol{S}}^{(k+1)} \right). \end{aligned}$$

With these choices, it can be shown that the Lagrange multipliers converge to a dual optimal solution; under mild conditions the primal variables also converge

to a primal optimal solution for (5) [29, 30]. The main utility in introducing the variables $\bar{\boldsymbol{L}}$ and $\bar{\boldsymbol{S}}$ is that it makes the solutions to subproblem (7) computable in nearly closed form. For example, (7) decouples into four independent minimizations, each of which has a closed form solution.

The major computation at each iteration consists in computing one partial singular value decomposition (SVD). The overall cost of the algorithm is proportional to the number of iterations times the cost of performing one partial SVD. It can be proved that ADMM converges at a rate of $O(1/k)$; for well-structured instances, it is often observed to outperform its worst case convergence theory.

5 Solution Properties and Errors

In general when only a portion of the entries in $\boldsymbol{A}_{full}$ are observed (i.e. $F < 1$), some subset of the entries in $\hat{\boldsymbol{A}}$ will be erroneous. Define $\boldsymbol{E} = \boldsymbol{A}_{full} - \hat{\boldsymbol{A}}$. The non-zero elements of $\boldsymbol{E}$ fall into two categories. *False negatives* are edges in G_{full} that are not present in $\hat{G}$. However, this is not great cause for concern since $\hat{G}$ still has fewer false negatives than G_{prm}, since the constraint in (1) ensures that $\hat{G}$ is a super-graph of G_{prm}. This implies that $\hat{G}$ has no more connected components than G_{prm}, inheriting its connectivity property. It also implies that the average path length in $\hat{G}$ is at least as small as that of G_{prm}. Obviously, since they share the same vertex set, $\hat{G}$ inherits the accessibility property.

On the other hand *false positives*–fictitious edges in $\hat{G}$–are of greater concern, since a robot executing such a path blindly, or at high speed, will experience a collision. In other applications this can be corrected for at run time using range or bump sensors.

There are a few general properties of the errors worth noting. First, there are no errors on the observation set Ω_{obs}, by virtue of the constraint in (1). From this it follows that there can be no errors on the support of $\boldsymbol{S}$ since $Supp(\boldsymbol{S}) \subset \Omega_{obs}$. To see this, consider that if S had a nonzero entry off of the observation set, setting the entry to zero would reduce the objective function without violating a constraint. These imply the following.

- The false negatives are a subset of $\neg Supp(\boldsymbol{L}) - \Omega_{obs}$.
- The false positives are a subset of $Supp(\boldsymbol{L}) - \Omega_{obs}$.

Of course, the structure of $\boldsymbol{L}$ and Ω_{obs} are influenced by the sample fraction F. There are three error regimes: *under-sampled*, *transition* and *over-sampled*.

During the *under-sampled regime* ($F < 2/n$), no extrapolation occurs–*i.e.* $\hat{\boldsymbol{A}} = P_{\Omega_{obs}}(\boldsymbol{A}) = \boldsymbol{A}_{prm}$. The diagonal constraint (3) effectively prevents extrapolation since the nuclear norm of the identity matrix is n. Up to one additional "1" can be added to each row (column) of $\boldsymbol{L}$ without increasing the nuclear norm so it is possible to satisfy the constraint (1) with $\boldsymbol{L} = P_{\Omega_{obs}}(\boldsymbol{A}_{full})$ and $\boldsymbol{S} = \boldsymbol{0}$. Just as in G_{prm}, there are no false positives but the fraction of false negatives is quite high:

$$Err = (1 - F)\frac{\|\boldsymbol{A}_{full}\|_0}{n^2}. \tag{8}$$

The *transition regime* ($F > 2/n$) begins after there are 3 or more observed "1"'s in any row (including the diagonal), causing L to begin extrapolating. When the extrapolations contradict an observed entry, S begins populating as well. The errors decrease monotonically from (8) to (9). The number of false negatives drops sharply while the number of false positives rises slightly.

During the *over-sampled regime* ($F > C \log^2 n/n$), the remaining errors come from the unsampled sparse entries. Let $\boldsymbol{L}_{full}$ and $\boldsymbol{S}_{full}$ refer to the decomposition when $F = 1$. Once F is above the transition threshold $\boldsymbol{L} = \boldsymbol{L}_{full}$ with high probability while $\boldsymbol{S} = P_{\Omega_{obs}}(\boldsymbol{S}_{full})$. Therefore in expectation

$$Err = (1 - F)\frac{\|\boldsymbol{S}_{full}\|_0}{n^2}, \tag{9}$$

with false positives and negatives on the unsampled negative and positive entries, respectively, of $\boldsymbol{S}_{full}$. Since no extrapolation occurs in the sparse term, the error only vanishes when $F = 1$.

6 Results

6.1 Examples

We consider four motion planning problems in this paper. For each problem G_{prm} was computed 10 times and for each of the resulting vertex sets, a ground truth visibility graph G_{full} and the corresponding adjacency matrix, $\boldsymbol{A}_{full}$, were computed by collision checking all possible edges using a resolution of 1% of the extent of C_{free}. All the results reported in this paper for a given problem are averaged over the 10 instances.

The first problem, *Simple 2-D* involves a point robot navigating in the environment shown in Fig. 2 and is primarily included because the results are easy to visualize. Next, three more realistic examples were generated using the Open Motion Planning Library (OMPL) [31]. *Twisty Cool* (Fig. 3 - left) involves an L-shaped object (red) traveling through a narrow passageway connecting the left and the right open regions. The passage can only be entered with a limited set of orientations and requires the object to rotate when partially though to exit the passage. Such problems are known to be challenging for sampling-base planners. *Twisty Cooler* (Fig. 3 - center) is an even more challenging scenario with multiple narrow passages. *Cubicles* (Fig. 3-right) involves a "T"-shaped robot (gold) moving through a multi-floor office building.

6.2 Model Applicability

First we support our claim that $\boldsymbol{A}_{full}$ has the low-rank plus structure posited in Sect. 2.3. In order to do so we set the sample fraction $F = 1$ and solve the optimization problem. Naturally $\hat{\boldsymbol{A}} = \boldsymbol{A}_{full}$ and $\boldsymbol{E} = \boldsymbol{0}$; however we are interested in the properties of the resulting $\boldsymbol{L}$ and $\boldsymbol{S}$.

In the case of *Simple 2-D* with $n = 200$, the results can be easily visualized. $\boldsymbol{L}$ is a rank-2 matrix. Figure 4 (left) illustrates the principle components of $\boldsymbol{L}$,

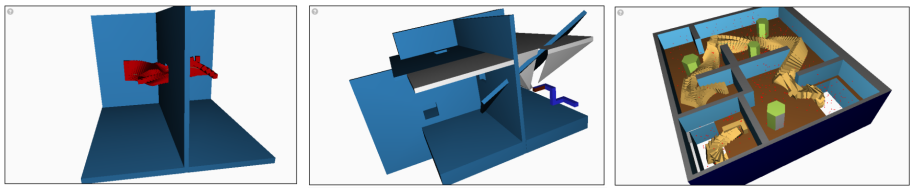


Fig. 3. Three examples from the Open Motion Planning Library: Twisty Cool (left); Twisty Cooler (center) and Cubicles (right).

which correspond to the edges within each large convex regions of C_{free}. Of course the edges connecting the two regions are missing from $\boldsymbol{L}$, and there are a few erroneous edges going through the small protrusion in the lower right corner. The *relative sparsity* of $\boldsymbol{S}$, defined as $\|\boldsymbol{S}\|_0/n^2$, is 0.0191, where 0.0121 of the entries are positive and 0.007 are negative. Fig. 4 (center) depicts the positive entries–edges that are added to $\boldsymbol{L}$. Many connect the two large convex regions. Fig. 4 (right) illustrates the negative entries, which are deleted edges that correct the false positives in $\boldsymbol{L}$ near the protrusion on the lower right.

The non-zero singular values of $\boldsymbol{L}$ (126 and 74) tell us there are two convex clusters of vertices in C_{free} containing 126 and 74 vertices. In higher dimensional examples this provides insight into the problem structure that is otherwise hard to glean. For Twisty Cool with 500 vertices, Figure 5 shows the original adjacency matrix on the left from which little can be discerned. The center panel shows the support pattern of $\boldsymbol{L}$ after the vertex labeling has been permuted so that identical row/columns are consecutively numbered. There are two clusters of size 247 and 229 (the two largest singular values). Another set of 20 vertices form the small interconnected group in the lower right corner. Their non-integer singular values (12.3 and 7.7) indicate that they cannot be successfully partitioned into clusters by the algorithm. They likely lie in the narrow passageway. Four additional vertices belong to no cluster (v_{490}, v_{494}, v_{499} and v_{500}); since their singular values are all 1. Their inclusion in $\boldsymbol{L}$ is an artifact of the diagonal constraint. The right panel shows $\boldsymbol{S}$ under the same permutation, where white entries are 1 and black entries are -1. We can see that most of the edges added by $\boldsymbol{S}$ connect v_{499} and v_{500} to the second large block (along outter edge of bottom right quadrant). v_{494} has a degree of 3–the smallest of any vertex.

A similar analysis was conducted for the remaining example problems, the results of which are summarized in Table 1. The effective rank (eliminating components whose singular values are ≤ 1) is around 5% of full and the sparsity is around 4%; suggesting that they all fit the low-rank plus sparse assumption. Our experiments suggest their relative values are not sensitive to n.

6.3 Completion Errors

Next we turn to our primary objective–completing the unknown entries in the adjacency matrix based on a small sample of observed entries. In these experi-

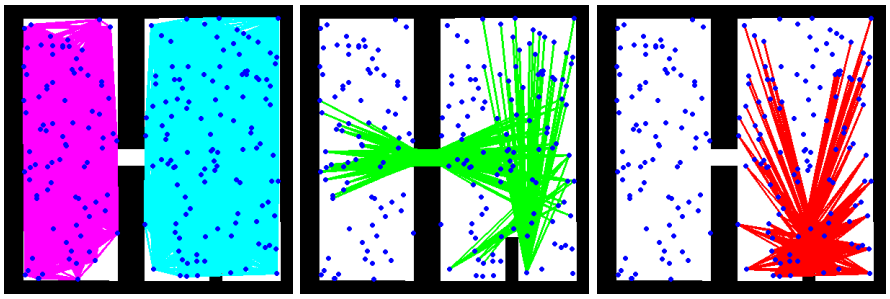


Fig. 4. The edges corresponding to the two principle components of $\boldsymbol{L}$ (left); the positive entries in $\boldsymbol{S}$ (center) and negative entries in $\boldsymbol{S}$ (right) for problem in Fig. 2 with 100% of the entries observed.

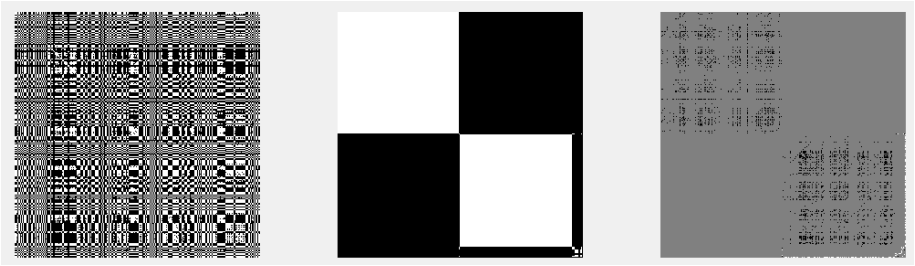


Fig. 5. The low-rank plus sparse decomposition for Twisty Cool (Fig.3-left). $\boldsymbol{A}_{full}$ (left) is difficult to interpret. While the permuted $\boldsymbol{L}$ (center) and $\boldsymbol{S}$ (right) provide insight into the the location of the vertices in C_{free}.

Example	**n**	**Eff. Rank**	**smallest s.v.**	$\lvert\boldsymbol{S}>0\rvert/n^2$	$\lvert\boldsymbol{S}<0\rvert/n^2$
Simple 2-D	200	2	74	0.0121	0.0071
Simple 2-D	500	2	151	0.0124	0.0068
Twisty Cool	250	4	6	0.0012	0.0399
Twisty Cool	500	4	7.7	0.0013	0.0370
Twisty Cooler	250	23	2	0.0152	0.0227
Twisty Cooler	500	29	2	0.0168	0.0238
Cubicles	250	16	2	0.0081	0.0312
Cubicles	500	23	2	0.0091	0.0323

Table 1. Properties of the decomposition when all entries in $\boldsymbol{A}_{full}$ are observed.

ments we ran the post-processing algorithm while varying the sample fraction F. Figure 6 shows the results for the Twisty Cool example with $n = 400$. The mean error rate is the solid blue trend line. The dashed blue line shows the predicted mean error rate during the under-sampled, transition and over-sampled phases from Sect. 5. The predicted end of the transition region is $F = 0.0508$; after this point the edge structure of $\hat{G}$ is better than 96% correct. Note that the mean false positive rate (shown in red), peaks at 0.0303 when $F = 0.0508$. Error plots are shown for Twisty Cooler (Fig. 8) and Cubicles (Fig. 9).

The dramatic reduction in path length for the Twisty Cool example is shown in Fig. 7 and is representative of the results for the other examples. At the end of the transition region, $F = 0.0503$, the average path length is 53% shorter in the post-processed graph, even after excluding the false positives. The number of connected components is identical in both graphs.

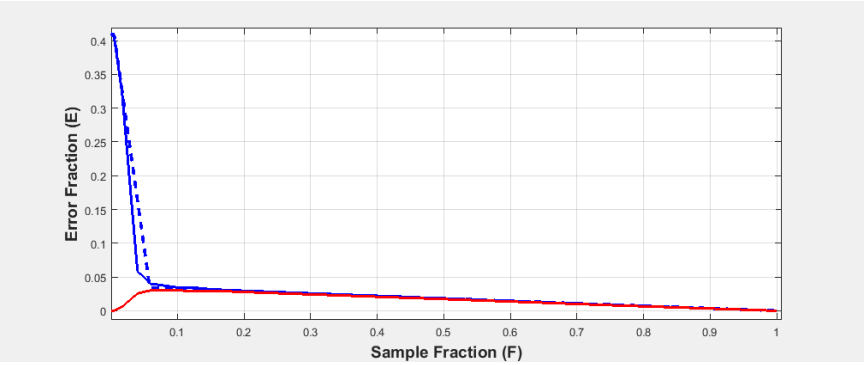


Fig. 6. The mean fraction of incorrect edges for Twisty Cool ($n = 400$) vs. the sample fraction F: experimental (solid blue); predicted (dashed blue); and false positives (red).

7 Connections to the Literature and Future Work

This paper introduced a post-processing technique for PRMs that uses convex optimization to estimate the status of unobserved graph edges without any additional collision checks. The post-processed graph, $\hat{G}$, inherits the connectivity and accessibility of G_{prm}. In a typical experiment, after directly checking only 5% of the possible edges, the structure of $\hat{G}$ was 96% accurate, with 53% shorter paths and 3% false positives. The practical utility of our algorithm is that the paths in the post-processed graph, $\hat{G}$, are dramatically shorter than those in G_{prm}– almost as short as in the full visibility graph which is prohibitively expensive to compute directly. A forthcoming study will extend the comparison to the path lengths produced by PRM*.

While few in number, the presence of false positives is troublesome. The ℓ^1 norm is symmetric, and hence does not introduce a clear preference towards

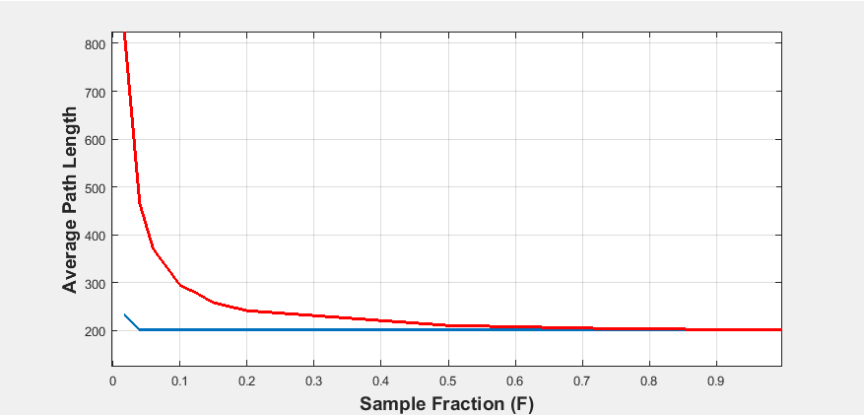


Fig. 7. The mean path length for Twistycool ($n = 400$): G_{prm} (upper red line) and $\hat{G}$ (lower blue line).

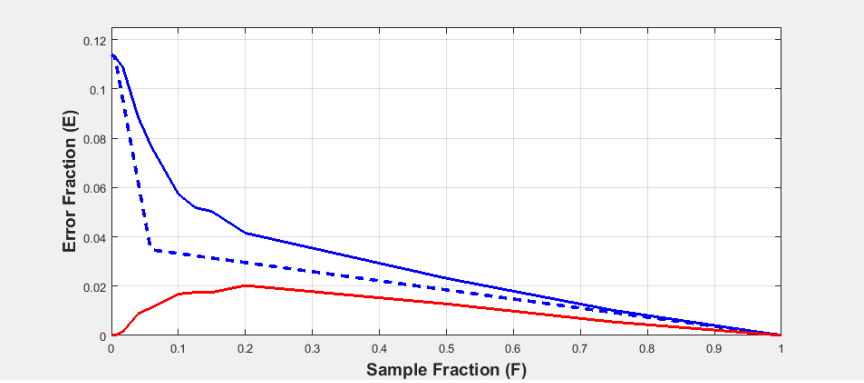


Fig. 8. The mean fraction of incorrect edges for Twisty Cooler ($n = 400$) vs. the sample fraction F: experimental (solid blue); predicted (dashed blue); and false positives (red).

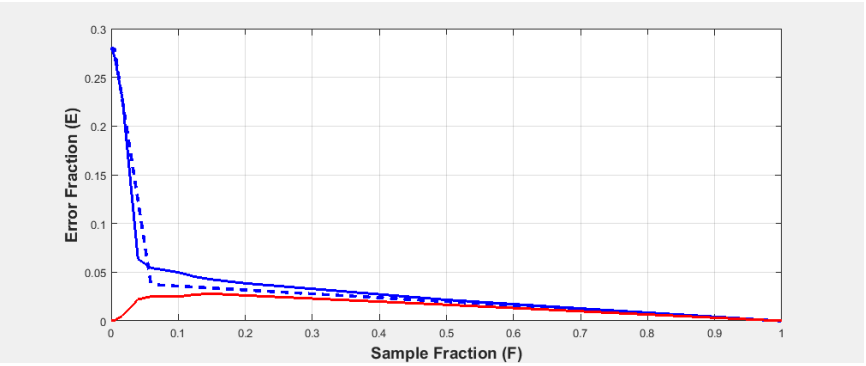


Fig. 9. The mean fraction of incorrect edges for Cubicles ($n = 400$) vs. the sample fraction F: experimental (solid blue); predicted (dashed blue); and false positives (red).

false positives or false negatives. Different types of errors may have very different practical implications. A preference for one type or the other can be introduced by replacing the ℓ^1 norm with an asymmetric penalty, in a manner similar to quantile regression [32]. Determining how to choose the penalty as a function of a user specified risk tolerance is another interesting topic for future work.

A rigorous assessment of the comparative computational savings of the proposed algorithm is a topic of future work. Recall that our proposed method essentially trades the cost of many additional collision checks for the cost of solving an interative convex optimization problem– motivated by the observation that collision checking the edges is the most expensive step in PRM [9]. Our post-processing algorithm is dominated by the time to compute a partial singular value decomposition.[4] Order analysis suggests that our approach is favorable. However, rigorously comparing the computation time of two radically different operations (collision checking vs. the optimization) is difficult because each is implemented in different programming languages, packages and machines; and depends on the complexity of the environment, the termination criteria and tolerances.

Overall, we believe the primary contribution of this work is in the novel application of tools from matrix completion to the application domain of robot motion planning. Our algorithm exploits the structure of the adjacency matrix which we hypothesize can be expressed as the sum of low-rank and sparse components. In essence this means that the adjacency matrix is highly redundant and can be encoded with very few bits of information, which can be captured with high probability by purely random sampling. We believe this provides an alternate explanation for how sampling-based planning methods are able to capture the connectivity of C_{free} with relatively few samples. Of course it is impossible to establish that this structure is representative of the set of *all possible* motion

[4] This cost can be further reduced by replacing ADMM with the Frank-Wolfe method, which only computes a single singular value/vector pair at each iteration [33].

planning problems. However, in the four examples considered here, the rank and sparsity of the resulting decomposition were less than 5% of their maximum values. In the same way that [10] argues that most motion planning problems of practical interest are expansive, we argue that the low-rank plus sparse structure is representative of common environments. In fact one area of future work is to relate the concept of $(\epsilon, \alpha, \beta)$-goodness to the properties of the low-rank and sparse matrices. If the examples here are indeed typical, it suggests motion planning problems are amenable to other techniques from that field as well. Results from the graph clustering literature concede that problems become difficult when the size of the smallest cluster approaches $\sqrt{n}$ and the sparsity approaches 1/3 [34]. By that measure, the graphs here are well outside of the hard regime (though the smallest cluster size is clearly the limiting factor); suggesting the use of convexified maximum-likelihood estimation.

The analysis of RPCA assumes the sparse entries are i.i.d Bernoulli random variables; however, in motion planning problems the location of these entries are likely to be concentrated in certain regions of the environment. We conducted preliminary experiments where we used the non-uniform sampling pattern induced by the common range limited variant of PRM and the results were not markedly different. However, adaptive sampling schemes are a promising area of future work.

Finally, a fundamental topic of interest in the matrix completion literature is *sample complexity*–the number of observations required to recover the low-rank term with high probability. The tightness of the $O(n \log^2 n)$ transition estimate of RPCA is an open question–the structure of adjacency matrices is not accounted for in the generic analysis [16]. Eliminating the second log factor would suggest a common range-limited PRM (requiring $O(n \log n)$ edge checks [9]) contains enough information, up to constant factors, to complete the low-rank term–eliminating the need for additional edge checks. This too is a topic of future work for us.

Acknowledgment

Joel Esposito is supported by ONR-N00014-13-0489; and John Wright is supported by ONR-N00014-13-0492. The authors would like to thank Lydia Kavraki and Mark Moll for providing the data sets used in this paper.

References

1. L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566580, June 1996.
2. E. Candès and B. Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9(6):717–772, 2009.
3. S. M. LaValle and J. J. Kuffner. Randomized kinodynamic planning. *International Journal of Robotics Research*, 20(5):378–400, May 2001.

4. D. Hsu, J.-C. Latombe, and R. Motwani. Path planning in expansive configuration spaces. *International Journal Computational Geometry & Applications*, 4:495–512, 1999.
5. J. Barraquand, L. Kavraki, J.-C. Latombe, T.-Y. Li, R. Motwani, and P. Raghavan. A random sampling scheme for robot path planning. In G. Giralt and G. Hirzinger, editors, *Proceedings International Symposium on Robotics Research*, pages 249–264. Springer-Verlag, New York, 1996.
6. A. Ladd and L. E. Kavraki. Measure theoretic analysis of probabilistic path planning. *IEEE Transactions on Robotics & Automation*, 20(2):229–242, 2004.
7. L. E. Kavraki, M. N. Kolountzakis, and J.-C. Latombe. Analysis of probabilistic roadmaps for path planning. *IEEE Transactions on Robotics and Automation*, 14(1):166–171, 1998.
8. S. M. LaValle, M. S. Branicky, and S. R. Lindemann. On the relationship between classical grid search and probabilistic roadmaps. *International Journal of Robotics Research*, 23(7/8):673–692, July/August 2004.
9. S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning. *International Journal of Robotics Research*, 30(7):846–894, 2011.
10. D. Hsu, J.-C. Latombe, and H. Kurniawati. On the probabilistic foundations of probabilistic roadmap planning. *International Journal of Robotics Research*, 25(7):627–643, July 2006.
11. E. Candès and T. Tao. The power of convex relaxation: Near-optimal matrix completion. *IEEE Transactions on Information Theory*, 56(5):2053–2080, 2010.
12. D. Gross. Recovering low-rank matrices from few coefficients in any basis. *IEEE Transactions on Information Theory*, 2010.
13. S. Neghaban and M. Wainwright. Estimation of (near) low-rank matrices with noise and high-dimensional scaling. *Annals of Statistics*, 39(2):1069–1097, 2011.
14. Y. Chen. Incoherence optimal matrix completion. *IEEE Transactions on Information Theory*, 61(5):2909–2923, 2015.
15. V. Chandrasekaran, S. Sanghavi, P. Parillo, and A. Willsky. Rank-sparsity incoherence for matrix decomposition. *SIAM Journal on Optimization*, 21(2):572–596, 2011.
16. E. Candès, X. Li, Y. Ma, and J. Wright. Robust principal component analysis? *Journal of the ACM*, 58(3), May 2011.
17. Daniel Hsu, Sham M Kakade, and Tong Zhang. Robust matrix decomposition with sparse corruptions. *IEEE Transactions on Information Theory*, 57(11):7221–7234, 2011.
18. X. Li. Compressed sensing and matrix completion with constant proportion of corruptions. *Constructive Approximation*, 37(1), 2013.
19. X. Yuan and M. Tao. Recovering low-rank and sparse components of matrices from incomplete and noisy observations. *SIAM Journal on Optimization*, 21(1):57–81, 2011.
20. Z. Lin, M. Chen, L. Wu, and Y. Ma. The augmented Lagrange multiplier method for exact recovery of corrupted low-rank matrices. Technical report, arXiv:1009.5055, 2009.
21. M. Davenport and J. Romberg. An overview of low-rank matrix recovery from incomplete observations. *IEEE Journal on Selected Topics in Signal Processing*, 2016.
22. N. Nilsson. A mobile automaton: An application of artificial intelligence techniques. In *Proceedings International Joint Conference on Artificial Intelligence*, May 1969.
23. B. Siciliano and O. Khatib, editors. *Springer Handbook of Robotics*. Springer, 2008.

24. J. H. Reif. Complexity of the mover's problem and generalizations. In *Proceedings IEEE Symposium on Foundations of Computer Science*, pages 421–427, 1979.
25. J. F. Canny. *The Complexity of Robot Motion Planning*. MIT Press, Cambridge, MA, 1988.
26. S. M. LaValle. *Planning Algorithms*. Cambridge University Press, Cambridge, U.K., 2006. Available at http://planning.cs.uiuc.edu/.
27. R. Glowinski and A. Marrocco. Sur l'approximation par él'ements finis d'ordre un, et la r'esolution, par p'enalisation-dualit'e d'une classe de probl'emes de dirichlet nonlin'eaires. *Revue Francaise d'Automatique, Informatique, Recherche Opérationnelle*, 9(2):41–76, 1975.
28. D. Gabay and B. Mercier. A dual algorithm for the solution of nonlinear variational problems via finite-element approximations. *Computers and Mathematics with Applications*, 2:17–40, 1976.
29. S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011.
30. J. Eckstein. Augmented lagrangian and alternating direction methods for convex optimization: A tutorial and some illustrative computational results. *RUTCOR Technical Report*, 2012.
31. I. A. Sucan, M. Moll, and L. E. Kavraki. The Open Motion Planning Library. *IEEE Robotics and Automation Magazine*, 19(4):72–82, 2012.
32. R. Koenker and K. Hallock. Quantile regression: An introduction. *Journal of Economic Perspectives*, 15(4):43–56.
33. C. Mu, Y. Zhang, J. Wright, and D. Goldfarb. Scalable robust matrix recovery: Frank-wolfe meets proximal methods. *SIAM Journal on Scientific Computing*, 2016.
34. Y. Chen, A. Jalali, S. Sanghavi, and H. Xu. Clustering partially observed graphs via convex optimization. *Journal of Machine Learning Research*, 15:2213–2238, 2014.