Prime Rigid Graphs and Multidimensional Scaling with Missing Data

Magnus Oskarsson, Kalle Åström and Anna Torstensson
Centre for Mathematical Sciences, Lund University, Sölvegatan 18, Lund, Sweden
{magnuso, kalle, annat}@maths.lth.se

***Abstract*—In this paper we investigate the problem of embedding a number of points given certain (but typically not all) inter-pair distance measurements. This problem is relevant for multi-dimensional scaling problems with missing data, and is applicable within anchor-free sensor network node calibration and anchor-free node localization using radio or sound TOA measurements. There are also applications within chemistry for deducing molecular 3D structure given inter-atom distance measurements and within machine learning and visualization of data, where only similarity measures between sample points are provided. The problem has been studied previously within the field of rigid graph theory. Our aim is here to construct numerically stable and efficient solvers for finding all embeddings of such minimal rigid graphs. The method is based on the observation that all graphs are either irreducibly rigid, here called prime rigid graphs, or contain smaller rigid graphs. By solving the embedding problem for the prime rigid graphs and for ways of assembling such graphs to other minimal rigid graphs, we show how to (i) calculate the number of embeddings and (ii) construct numerically stable and efficient algorithms for obtaining all embeddings given inter-node measurements. The solvers are verified with experiments on simulated data.**

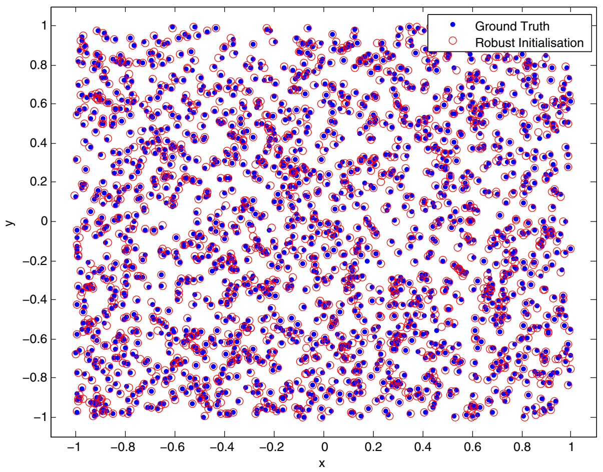

Fig. 1. The figure shows the result of running our robust initialization on a large problem with missing data. The resulting point positions are shown as red circles. The solution has been registered to the ground truth solution in blue. One can see that there is little dicrepancy between the two solutions.

I. INTRODUCTION

In this paper we will study the problem of determining positions $\mathbf{r}_i, i = 1, \ldots, n$ given distances $d_{ij} = |\mathbf{r}_i - \mathbf{r}_j|$, for a subset of point pairs $(i, j) \in E$. This is a classic problem that has intrigued engineers and researchers for centuries. One of the first results, the rigidity theorem, is due to Cauchy [1]. Originally the problem had implications in the construction of trusses, but today there are several practical applications where this problem is relevant, e.g. in

- Anchor-free node localization using distance measurements. See e.g. [2].
- Autonomous vehicle formation, see e.g. [3].
- Shape determination from inter-point distances, e.g. estimating molecule shape using inter-atom distance measurements.
- Anchor-free sensor network mapping, see e.g. [4].

If the distances for all pairs of points are known the problem is known as multidimensional scaling or MDS [13]. In this paper we will instead consider the problem when only a subset of the distances are known, i.e. MDS with missing data. We will start by giving a theoretical basis, which will then be used in order to construct efficient solvers. The result of running our solver on a large scale problem with missing data is shown in figure 1.

The problem has been studied in graph theory. One observation is that the problem depends on the number of nodes n and which distance pairs (i, j) that have been measured. This can be conveniently represented as a graph (V, E), where V is the set of n vertices or nodes $V = \{1, \ldots, n\}$ and $E \subseteq V \times V$ is the set of edges. The problem that we want to solve is that of calculating all possible embeddings, i.e. all possible sets of points, that fulfill the inter-node distances d_{ij} for $(i, j) \in E$. We will formulate this problem in a slightly different way. First we need a more formal definition of what we mean by an embedding.

Definition 1.1: A graph $G = (V, E)$ together with n points $\mathbf{p}_i, i = 1, \ldots, n$ in d-dimensional Euclidean space is called an embedding of G in $\mathbb{E}^d$. We denote this embedding $G(p)$.

Now we can formulate our problem in the following way:

Problem 1.1: Given an embeddning of a graph G determine all node positions $\mathbf{r}_i, i = 1, \ldots, n$, such that

$$d_{ij} = |\mathbf{p}_i - \mathbf{p}_j| = |\mathbf{r}_i - \mathbf{r}_j|, \quad \forall (i, j) \in E.$$

The problem is invariant under Euclidean transformations and mirroring. Factoring out such natural Gauge freedom, a particular embedding problem can have either a finite or an infinite number of solutions. In the former case the embedding $G(p)$ is called rigid, and in the latter case flexible. Moreover an embedding is called globally rigid if there is exactly one solution to the embedding problem. (Note that there is always at least one solution given by $\mathbf{r}_i = \mathbf{p}_i$.)

In [5] the authors show that being flexible or rigid is a property of the graph, and not the embedding, in the

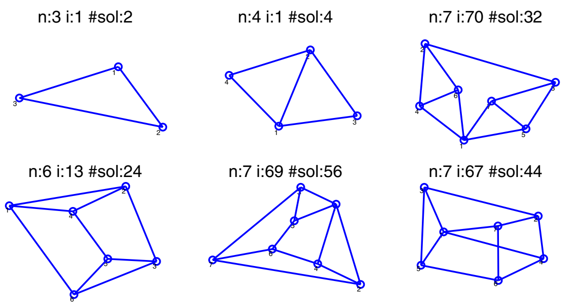

Fig. 2. Minimal and prime rigid graph $G_{3,1} = P_1$ with three nodes (top left), minimal but non-prime rigid graph $G_{4,1}$ with four nodes (top middle), the Moser spindle (top right), a three edge glue E_1 (bottom left), a one hinge and two node glue E_2 (bottom middle) and a four node extension E_3 (bottom right). The number of embeddings for each problem is also given.

sense that for nodes in generic positions the rigidity is well defined. However, for a rigid graph there might be degenerate configurations of node positions that cause the construction to be flexible and vice-versa, but this set of configurations has Lebesgue measure zero. Subsequently we will therefore talk about flexible or rigid graphs, meaning that for a generic choice of points p, the embedding problem for $G(p)$ is flexible or rigid respectively.

Our goal is to characterize all rigid graphs and find their possible embeddings. In order to do this it is sufficient to study the minimal rigid graphs. (Minimal in the sense that the removal of any edge detroys the rigidity.) This is based on the observation that any rigid graph contains a minimal rigid graph, obtained by removing edges in the original graph. By solving the embedding problem for all minimal rigid graphs one may thus solve this problem for any rigid graph by checking if the solutions for the contained minimal rigid graph also satisfies the conditions given by the removed edges.

Therefore we will describe a framework that gives us a comprehensive description of the minimal rigid graphs, in terms of number of solutions and practical solution algorithms. This in turn gives us tools to:

- Determine whether a given problem is well posed.
- Construct minimal solvers that are basic components in robust estimation algorithms, such as RANSAC [6], for large scale problems.

For more background see [3], [7]. In [8] the authors consider the rigidity question for full bipartite graphs. The results are for general dimensionality. One result is that for planar graph all full bipartite graphs containing $K_{3,3}$ are rigid and in 3D all full bipartite graphs containing either $K_{4,6}$ or $K_{5,5}$ are rigid. Here $K_{m,n}$ denotes the complete bipartite graph on m and n vertices, i.e. the bipartite graph $G = (U, V, E)$, where U and V are disjoint sets of size m and n, respectively, and E connects every vertex in U with all vertices in V. There has been work on calculating the number of embeddings for different minimal problems, see [9], [10]. We have some contradicting results to those given in [10], see section IV for more details.

Most of the general theory that we present in this paper is valid for any dimension, but for the specific solvers we will concentrate on the planar case, i.e. the unknown node positions $\mathbf{r}_i$ are points in $\mathbb{R}^2$. We will in the next section describe how the minimal planar rigid graphs can be constructed in a way that gives us ways of constructing solution algorithms as well as determining the number of (complex) embeddings for different such graphs. In section IV we will show results on constructions for up to ten nodes and in section V we will give experimental results that validate our approach on synthetic data.

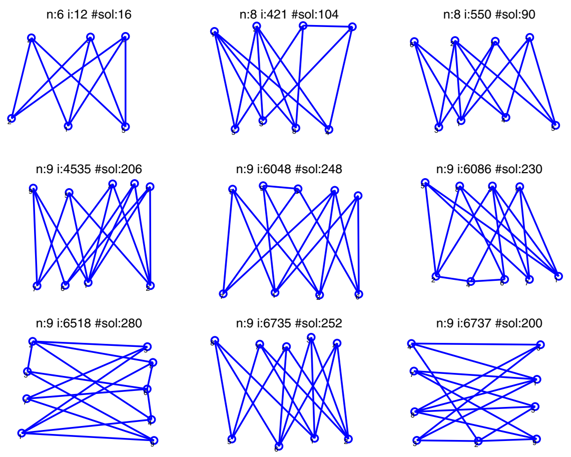

Fig. 3. These 9 prime rigid graphs together with the prime graph in figure 2 are all the prime rigid graphs P_1 to P_{10} for up to 9 nodes. The number of embeddings for each is also shown.

II. The planar embedding problem

A graph G is *minimally rigid* if G is rigid and $G - e$ is not rigid for any edge e of G. In $\mathbb{R}^2$ there is an equivalent formulation due to Laman [11] namely: A graph G with n nodes and m edges is *minimally rigid* if $m = 2n - 3$ and for any vertex induced subgraph of G with $k \geq 2$ nodes and l edges we have $l \leq 2k - 3$. In the planar case the minimally rigid graphs are also called *Laman graphs*.

Note that the definition of a Laman graph says that the system of equations in the embedding problem for such a graph is balanced in terms of number of equations and unknowns and each subsystem corresponding to a subgraph is either balanced or underdetermined. We will use the Laman formulation to make the following definition:

Definition 2.1: A graph G is a *Prime Rigid Graph* if it is minimally rigid and for any subgraph of G with $k \geq 3$ nodes and l edges we have $l < 2k - 3$.

Thus in a prime rigid graph all subsystems corresponding to vertex induced subgraphs (on at least three vertices) are underdetermined. This implies that all those subgraphs are flexible, since it follows from the Laman characterization that a rigid graph cannot correspond to a underdetermined system.

For the Laman graphs there is a recursive way due to Henneberg [12] of constructing all the graphs using two types of extensions, Henneberg I and Henneberg II. One starts with the graph consisting of two nodes joined by one edge. In the Henneberg I extension one adds one new node and joins this with two edges to the old set of nodes. In the Henneberg II extension one removes one edge (i, j) among the previous ones, and connect a new node to the two nodes i and j. The new node is also connected to a third node among the previous ones.

The Henneberg I construction can be used constructively in the following two senses. (i) If the number of embeddings is known for a graph G, then the number of solutions after a Henneberg I extension is twice as many. This is because the possible positions of the added node are the intersection points of the two circles describing its distances to its neighbours. For a generic position of the added node there are two points of intersection. (ii) if an algorithm for finding all embeddings given distances is given for a graph G, then it is easy to extend this algorithm to the Henneberg I extension of G.

For the Henneberg II extension, less is known, both in terms of multiplicity of solutions and in terms of practical algorithms for determining all possible embeddings.

We will now show how all minimally rigid graphs can be constructed using the prime and minimal graphs as basic building blocks and combining them in two different ways. As for the case of Henneberg extensions, these extensions do not generate unique ways of constructing the minimal graphs. But as opposed to the Henneberg extensions they let us keep track of the number of solutions as well as being able to construct solvers for the different minimal problems. We will denote the ith minimal graph with n nodes by $G_{n,i}$.

A. *Extending a minimal graph with another minimal graph*

The first type of extension is as follows. Let g_1 and g_2 be minimal graphs with n_1 and n_2 embeddings respectively. A new minimal graph can be constructed by removing an edge (i, j) in g_1 and inserting g_2 so that two of the nodes in g_2 are merged with the nodes i and j respectively. The resulting graph g is minimal and has $n_1 n_2$ solutions.

If there are algorithms for finding all embeddings given distance measurement both for g_1 and g_2, then an algorithm for finding all embeddings for g can be constructed as follows. Given length measurements for g, first solve for all n_2 embeddings for the graph g_2. For each such solution calculate the distance between nodes i and j. Use this distance together with the other distances for g_1 and the algorithm for g_1 to calculate all embeddings for g_1. For each of the n_2 embeddings for g_2 there are in general n_1 embeddings for g_1. Thus the algorithm outputs all $n_1 n_2$ embeddings for g.

Note that in the special case of g_2 being the triangle P_1 (and thus $n_2 = 2$) one obtains the Henneberg I extension.

B. *Joining a number of minimal graphs, called a* Glue

By connecting a number of minimal graphs we can get a new minimally rigid graph. The connections between the different parts can be achieved by a combination of:

- Merging two or more nodes from the different graphs into one node, called a *Hinge*.
- Adding a number of edges between the different graphs.
- Adding a number of new nodes with connections to the different graphs.

We will consider the case of adding a number of nodes to one minimal graph as a special case of this extension. This means that a Henneberg I extension also can be described as a one node extension to a minimal graph. We will in section III describe specific types of possible extensions that generate all minimal graphs up to order 8, as well as the prime rigid graphs up to order 9.

In order to be able to count the number of embeddings and constructing solution algorithms for these cases we need to solve for the basic building blocks, i.e. the prime rigid graphs as well as constructing solvers for the different extensions.

There are several interesting uses of these ideas. Even with only a few solved prime rigid graphs and extensions, it is possible to automatically generate solvers for a large class of minimal graphs. For these graphs we also obtain the number of embeddings. Even if we only obtain the number of embeddings for a number of prime rigid graphs and/or extensions, it is still possible to use these as building blocks in order to determine the number of embeddings for the reachable minimal graphs. A third use is to determine the minimal graphs that cannot be solved by a set of building blocks. These graphs are important to provide the building blocks that are missing. The process is similar to that of determining primes using the sieve of Eratosthenes, i.e. the smallest prime rigid graphs and extensions can be used to solve many minimal graph problems. The remaining graphs are either prime or contain a prime sub-graph with a new type of extension. One question is how many prime rigid graphs we need to solve in order to solve all minimal graphs. The following theorem gives the answer.

Theorem 2.1: There are infinitely many planar prime rigid graphs.

Proof: For any number $n > 2$ we can construct the following bi-partite graph G with $2n$ nodes. Denote the first n nodes by u_i and the last n nodes by l_i. From node u_i we draw an edge to nodes $(l_i, l_{i+1}, l_{i+2}, l_{i+3})$ for $i = 1 \ldots n-3$. From node u_{n-2} we draw edges to nodes (l_{n-2}, l_{n-1}, l_n). From node u_{n-1} we draw edges to nodes l_{n-1}, l_n, l_1 and from node u_n we draw edges to nodes (l_n, l_1, l_2). An example with $n = 4$ is shown in Figure 3. For the full graph we have $4 \cdot (n-3) + 3 \cdot 3 = 4n - 3$ edges which equals the $2 \cdot 2n - 3 = 4n - 3$ unknowns. By simple inspection it is clear that all sub-graphs will be under-determined. This leads to that a construction of this type, for any n, will be prime. ■

III. A NUMBER OF SPECIFIC BASIC BUILDING BLOCKS

In order to build up the minimal graphs we need to first of all find all prime rigid graphs. This can be done by generating all minimal graphs up to some order recursively using Henneberg I and II extensions. We can then test whether they are prime using definition 2.1 directly by constructing all sub-graphs. The number of prime rigid graphs for the different number of nodes is given in Table I and all the prime rigid graphs of order up to 9 are shown in Figure 3. We will number the prime graphs consecutively and denote prime graph number i with P_i. We will further number all the minimal graphs so that $G_{n,i}$ denotes the ith minimal graph with n nodes. More details are given in the next section.

In addition to finding the prime rigid graphs, we also need to construct extensions. In this section we will study a number of specific instances of extensions of type two.

A. Three Edge Glue of two Minimal Graphs (E_1)

Given two minimal graphs g_1 and g_2. By connecting these two minimal graphs with three edges $(i_k, j_k), k = 1, 2, 3$, where i_k is a node of g_1 and j_k is a node of g_2 one obtains a minimal graph, cf. bottom left of Figure 2. We will denote this operation by E_1. The multiplicity or the number of solutions for this glue is $e_1 = 6$. Similarly given algorithms for finding the embeddings for g_1, g_2 and the glue E_1 it is straightforward to construct an algorithm for the resulting graph g. The number of embeddings is now $n_1 n_2 e_1$. An example of glueing together a three node minimal graph with a four node minimal graph is given in Figure 2.

B. One Hinge and two Node Glue of two Minimal Graphs (E_2)

Another interesting way of glueing together two minimal cases is as follows. Given two minimal graphs g_1 and g_2. By connecting these two minimal graphs at one node (like a hinge) and then connecting each graph with two nodes to two new nodes according to Figure 2 one obtains a minimal graph g. We will denote this operation by E_2. The multiplicity or the number of solutions for this glue is $e_2 = 14$. Similarly given algorithms for finding the embeddings for g_1, g_2 and the glue E_2 it is straightforward to construct an algorithm for the resulting graph g. The number of embeddings is now $n_1 n_2 e_2$. In the figure we see this applied to two triangles resulting in $2 \cdot 2 \cdot 14 = 56$ solutions.

C. Four Node Extension of one Minimal Graph (E_3)

The following construction is an irreducible extension of one minimal graph. Given one minimal graphs g_1. By connecting the graph with four edges to four new nodes according to according to Figure 2 one obtains a minimal graph g. We will denote this operation by E_3. The multiplicity or the number of solutions for this glue is $e_3 = 14$. Similarly given algorithms for finding the embeddings for g_1 and the extension E_3 it is straightforward to construct an algorithm for the resulting graph g. The number of embeddings is now $n_1 e_3$.

IV. Study of minimal graphs up to order 10

In this paper we have generated and studied all minimal graphs up to $n = 10$ nodes. These minimal graphs can be generated by Henneberg I and Henneberg II constructions starting with either $n = 2$ or $n = 3$. For all of these graphs, we have tested if the problems are prime. If not we have tested if they are reachable with the building blocks from the previous sections, and if so, we have automatically constructed algorithms for finding all embeddings. For problems that are not reachable with the building blocks above, we have also generated integer problems and investigated the number of embeddings over $\mathbb{Z}_p$, where p is a large prime number. This is often a good indication of the number of embeddings over $\mathbb{C}$.

A summary of the investigation is as follows.

TABLE I. The table shows for each number of nodes, the number of edges, the number of minimal graphs, the number of prime rigid graphs, the range of the number of embeddings, the number of graphs that can not be reached using Henneberg I extensions, and the number of graphs that can not be reached using primes and our first three extensions.

n	k	# minimal	# prime	# embeddings	H_1	primes + E_1, E_2, E_3
3	3	1	1	2	1	0
4	5	1	0	4	0	0
5	7	3	0	8	0	0
6	9	13	1	16,24	2	0
7	11	70	0	32, 44, 48, 56	4	0
8	13	608	2	64, …, 136	32	11
9	15	7222	6	128, …, 312	265	126
10	17	110132	44	256, …, 672	3237	1874

For $n = 3$ nodes. There is only one minimal graph with 3 nodes. This graph, the complete graph K_3 is minimal and prime. It will be denoted P_1 in the sequel. It has a unique solution (or two solutions if one counts the mirrored solution as different, cf. [13]. Thus the number of embeddings $p_1 = 2$.

The case of $n = 4$ nodes. With four nodes, there are 6 possible edges. The potential minimal graphs have 5 edges. There is only one such graph. It is minimal, but not irreducible. It contains P_1 as a sub-graph and can be generated by the first construction, here denoted P_1^2, with $p_1^2 = 4$ number of embeddings.

The case of $n = 5$ nodes. With five nodes, there are 10 possible edges. The potential minimal graphs have 7 edges. There are four such graphs. Of these four graphs three are minimal (but not prime) and can be constructed from P_1 as P_1^3 with $p_1^3 = 8$ embeddings.

The case of $n = 6$ nodes. There are 13 minimal rigid graphs with 6 nodes. Eleven of these are of type P_1^3. They are thus not prime and are easily solved with $p_1^3 = 8$ embeddings. One of the graphs, the so called Desargues graph consists of two triangles joined by three edges. It is thus of type $P_1^2 E_1$. The algorithms correctly outputs $p_1^2 e_1 = 24$ solutions. The last graph is a new prime rigid graph, called $K_{3,3}$. Here it will be denoted P_2. It has $p_2 = 16$ embeddings. An algorithm for finding these can be found in [14].

The case of $n = 7$ nodes. There are 70 minimal rigid graphs with 7 nodes. Of these 64 are of type P_1^5 with $p_1^5 = 32$ solutions. Two graphs are of type $P_2 P_1$ with 32 solutions. One is of type $P_1^3 E_1$ with 48 solutions. One is of type $P_1^2 E_2$ with $p_1^2 e_2 = 56$ solutions and one is of type $P_1 E_3$ with $p_1 e_3 = 28$ solutions. The solvers for E_2 and E_3 are to the best of our knowledge new. There are no prime rigid graphs of order 7.

With $n = 8$ nodes, it is getting difficult to describe all cases. There are 608 graphs of which 526 have 64 solutions and are easy to solve with combinations of P_1 and P_2. There are 46 graphs with 96 solutions which involve E_2 and P_1. There are several new types of extensions and there are 2 new prime rigid graphs.

Using our constructive solution algorithms we can find the number of embeddings of all graphs that can be reached using the building blocks described in sections II and III. We have done this for all graphs up to order 10 and the number of graphs that we can not reach is given in Table I. For the remaining graphs we have investigated the number of solutions by generating realisations of problems with integer distances. We have then found the Grobner basis for the ideal generated by the equations over $\mathbb{Z}_p$. This gives with high probability the dimension of the underlying problem. We have used Macaulay 2 for the calculations, cf. [15]. This gives the resulting number of embeddings shown in Table I. We have some contradicting results compared to [10] where the upper bound for 8 nodes was given as 128 where we have a minimal problem that yields 136 solutions.

V. Experimental validation

We have conducted a number of experiments on a number of different levels to validate our methods and solvers. Firstly we have tested the numerical stability of the minimal solvers. We then tested how well the minimal solvers behaved in the presence of noise and finally we contructed a small system for larger problems to show how our solvers can be used to bootstrap robust estimation in the precence of noise, outliers and missing data. The system results can be seen in figure 1

A. Experiments on initial solvers

We have implemented several solver building blocks. The solver for the triangle is straightforward. The solver for the bipartite graph $K_{3,3}$ was done in another context in [14]. We could not find a solver for the three-edge glue, E_1, in the literature, so we developed a novel one based on the action matrix method, [16], [17]. Also for the one-hinge and two node glue of two minimal graphs, E_2, we developed a novel solver.

Furthermore we developed a framework to automatically generate solvers for all minimal graphs that can be constructed from these building blocks. The solvers were tested in the following way. For each tested graph, we generated random configurations $\mathbf{r}_0$. Then the distance measurements $D_0 = \{d_{ij}\}$ where calculated and given as input to the solver. The solver then generated all n solutions $\hat{\mathbf{r}}_1, \ldots, \hat{\mathbf{r}}_n$, where n is the multiplicity of solutions for the minimal graph. The n different reconstructions were checked both to determine how well the distances from the embeddings $D(\hat{\mathbf{r}}_k)$ matched the specified distances D_0. Ideally these should all be very small. It was also investigated if the random configuration $\mathbf{r}_0$ was close to anyone of the n solutions. Ideally it should be identical to one of the solutions. We repeated this experiment 1000 times. In Figure 4 is shown the histogram of the 10-log of residuals in blue and of the reconstruction error in red. This is done for four minimal graphs, top-left the triangle $G_{3,1}$, top right the graph $G_{4,1}$, bottom-left the prime rigid graph $G_{6,12}$ and bottom-right a more complicated compound graph $G_{9,1}$.

B. Experiments on building blocks

To test the behaviour of our solvers in the presence of noise we conducted an experiment in the following way. We generated a full small problem, of say seven nodes. The x and y coordinates of the points were drawn from a uniform distribution on $[-1, 1]$. To the distance matrix we added Gaussian noise with zero mean. We then extracted a random minimal subproblem (with the same number of nodes, but fewer edges) and solved this with corresponding minimal

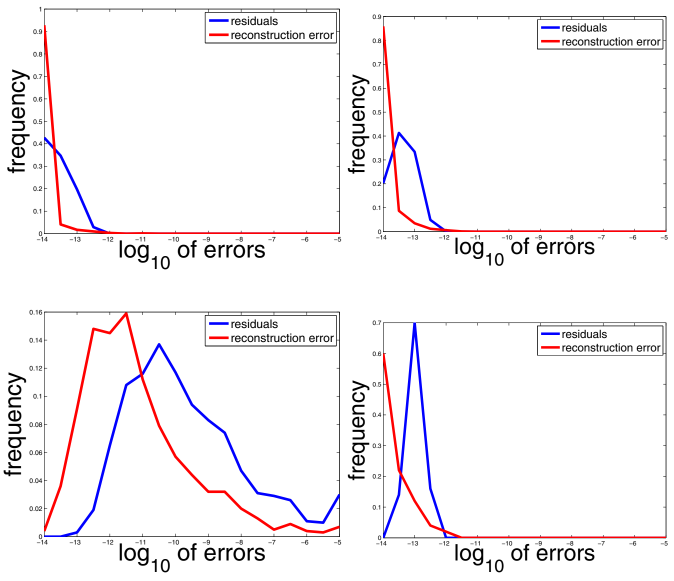

Fig. 4. Histogram of residuals (blue) and reconstruction error (red) for 1000 random configurations of (i - top-left) the triangle $G_{3,1}$, (ii - top right) the graph $G_{4,1}$, (iii - bottom-left) the prime rigid graph $G_{6,12}$ and (iv - bottom-right) a more complicated compound graph $G_{9,1}$.

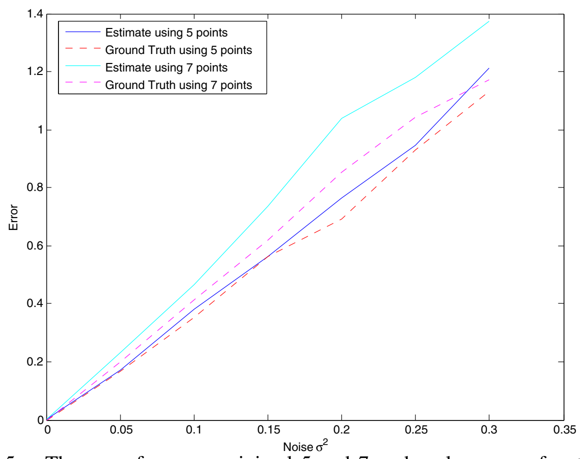

Fig. 5. The error from our minimal 5 and 7 node solvers as a function of the variance of the added noise. Also shown is the corresponding error for the ground truth solutions.

solver. We then took each of the solutions and checked how well they fit the removed edges, i.e. we measured the norm of the distance matrix for the full problem. One of the solutions should fit all the distances well. In figure 5 the resulting norm of the distance matrix for the best solution is shown as a function of the variance of the added noise. This was done for $n = 5$ nodes, so in this case there were three possible minimal subproblems. The norm of the distance matrix for the ground truth solution is also shown. We see that our solvers are close to linear for these amounts of noise. They also follow the ground truth errors well. The results for $n = 7$ nodes are also shown. In this case there are 13 possible minimal subproblems. The results for this case are similar to the five node case.

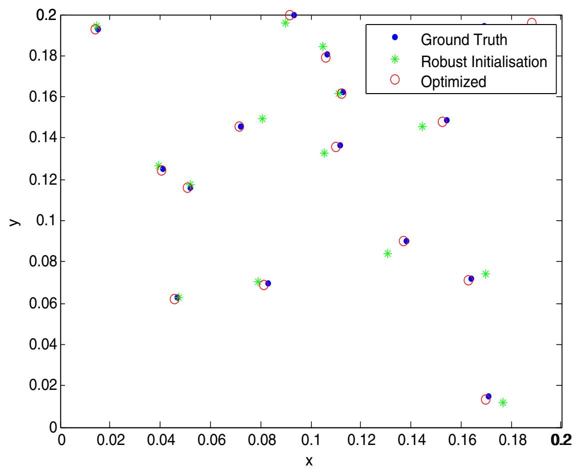

Fig. 6. A magnification of part of the solutions from figure 1. The green stars show the result of our robust initialization. The red circles denotse the result after additional nonlinear optimization. The solutions are registered to the ground truth solution in blue.

C. Experiments on system level

To test our solvers on a systems level we constructed the following experiment. We generated a full large problem, with e.g. 2000 nodes. The x and y coordinates of the points were drawn from a uniform distribution on $[-1, 1]$. To the distance matrix we added Gaussian noise with zero mean. We also added outliers in the distance matrix, i.e. random distances $[0, 2\sqrt{2}]$. We also removed a number of distances to simulate missing data. We then used a RANSAC framework to estimate the full solution. We randomly tested a small minimal problem. with e.g. 6 nodes, and checked how well other distances fit this model. When we found a good initial solution we used our algorithms for extensions to extend this initial solution to the full solution. In the top of figure reffig:regsolution the resulting solution for one case is shown registered to the ground truth solution. In this case we had 15% outliers and 15% missing data and the added noise had standard deviation 0.1. We see that the solution fits the ground truth quite well. We also used this solution as initialisation for a non-linear optimization of the distance matrix. We simply used a Newton method to minimize the norm of the difference of the estimated distance matrix with the given one. The resulting solution is shown in the bottom of figure 1. In figure 6 a part of the solution is shown in detail. In figure 7 the convergence behaviour of the non linear optimization is shown as a function of iteration. Also shown is the behaviour if we choose a random initial solution which doesn't converge at all. A comparison to the groud truth is also shown.

VI. Conclusions

In this paper we have introduced the notion of prime rigid graphs and extensions of minimal rigid graphs. The technique enables us to precisely characterize the number of possible embeddings of rigid graphs and to automatically generate solution algorithms for finding all possible embeddings. Using simulated data we have shown that these algorithm are numerically stable, and can be used as building blocks for robust estimation systems.

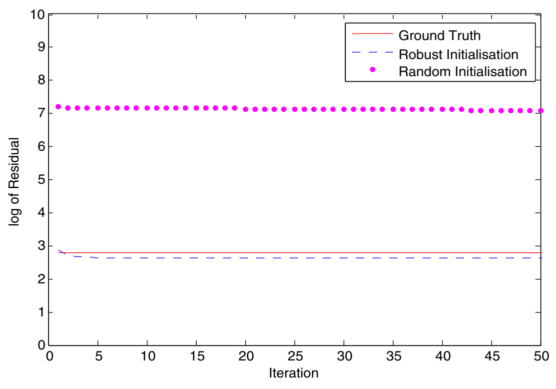

Fig. 7. The convergence behaviour for the optimization as a function of iteration. Also shown is the ground truth error.

References

[1] A. Cauchy, "Recherche sur les polyedres - premier mémoire," *Journal de l'Ecole Polytechnique*, no. 9, pp. 66–86, 1813.

[2] M. S. Brandstein, J. E. Adcock, and H. F. Silverman, "A closed-form location estimator for use with room environment microphone arrays," *Speech and Audio Processing, IEEE Transactions on*, vol. 5, no. 1, pp. 45–50, 1997.

[3] B. Anderson, C. Yu, B. Fidan, and J. Hendrickx, "Rigid graph control architectures for autonomous formations," *Control Systems, IEEE*, vol. 28, no. 6, pp. 48–63, 2008.

[4] T. Eren, O. Goldenberg, W. Whiteley, Y. R. Yang, A. S. Morse, B. Anderson, and P. Belhumeur, "Rigidity, computation, and randomization in network localization," in *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, vol. 4. IEEE, 2004, pp. 2673–2684.

[5] L. Asimow and B. Roth, "The rigidity of graphs, ii," *Journal of Mathematical Analysis and Applications*, vol. 68, no. 1, pp. 171–190, 1979.

[6] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–95, 1981.

[7] T.-S. Tay and W. Whiteley, "Generating isostatic frameworks," *Structural Topology 1985 núm 11*, 1985.

[8] E. D. Bolker and B. Roth, "When is a bipartite graph a rigid framework," *Pacific J. Math*, vol. 90, no. 1, pp. 27–44, 1980.

[9] R. Steffens and T. Theobald, "Mixed volume techniques for embeddings of laman graphs," *Computational Geometry*, vol. 43, no. 2, pp. 84–93, 2010.

[10] I. Emiris, E. Tsigaridas, and A. Varvitsiotis, "Algebraic methods for counting euclidean embeddings of rigid graphs," in *Graph Drawing*. Springer, 2010, pp. 195–200.

[11] G. Laman, "On graphs and rigidity of plane skeletal structures," *Journal of Engineering mathematics*, vol. 4, no. 4, pp. 331–340, 1970.

[12] L. Henneberg, *Die graphische Statik der starren Systeme*. BG Teubner, 1911, vol. 31.

[13] G. Young and A. Householder, "Discussion of a set of points in terms of their mutual distances," *Psychometrika*, vol. 3, no. 1, pp. 19–22, 1941.

[14] H. Stewénius, "Gröbner basis methods for minimal problems in computer vision," Ph.D. dissertation, Lund University, APR 2005.

[15] D. Grayson and M. Stillman, "Macaulay 2," Available at http://www.math.uiuc.edu/Macaulay2/, 1993-2002, an open source computer algebra software. [Online]. Available: http://www.math.uiuc.edu/Macaulay2/

[16] M. Byröd, K. Josephson, and K. Åström, "Fast and stable polynomial equation solving and its application to computer vision," *Int. Journal of Computer Vision*, vol. 84, no. 3, pp. 237–255, 2009.

[17] D. Cox, J. Little, and D. O'Shea, *Using Algebraic Geometry*. Springer Verlag, 1998.