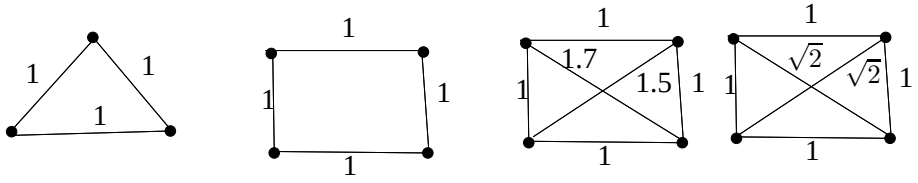

Figure 1: (*left*) Wellconstrained System; *middle left:* underconstrained system; *middle right:* overconstrained system; (*right*) consistently overconstrained system

1 Rigidity

Geometric constraint systems have been studied in the context of variational constraint solving in CAD for nearly 2 decades. [65, 40, 64, 50, 34, 36, 39, 48, 13, 60, 31, 5, 49, 61, 17, 62, 63] [6, 11, 28, 29, 20, 21, 1] [23, 26, 27, 25] [30, 19, 25, 14, 15, 43, 3, 4] [44, 7, 42, 47, 2, 45, 46] [18, 37, 41, 52]. For recent reviews of the extensive literature on geometric constraint solving more elaborate descriptions and examples for the definitions below, see, e.g, [23, 33, 12, 53].

1.1 Definitions

A *geometric constraint system* consists of a finite set of primitive geometric objects such as points, lines, planes, conics etc. and a finite set of geometric constraints between them such as distance, angle, incidence etc. The constraints can usually be written as algebraic equations and inequalities whose variables are the coordinates of the participating geometric objects. For example, a distance constraint of d between two points (x_1, y_1) and (x_2, y_2) in 2D is written as $(x_2 - x_1)^2 + (y_2 - y_1)^2 = d^2$. In this case the distance d is the *parameter* associated with the constraint. Most of the constraint solvers so far deal with 2D constraint systems. With the exception of work [22, 24, 26, 27], [25], [18, 38, 37, 41, 51, 58, 56, 57], related to the FRONTIER geometric constraint solver [52], to the best of our knowledge, work on stand-alone 3D geometric constraint solvers is relatively sparse [7, 42].

A *solution or realization* of a geometric constraint system is the (set of) real zero(es) of the corresponding algebraic system. In other words, the solution is a class of valid instantiations of (the position, orientation and any other parameters of) the geometric elements such that all constraints are satisfied. Here, it is understood that such a solution is in a particular geometry, for example the Euclidean plane, the sphere, or Euclidean 3 dimensional space. A constraint system can be classified as *overconstrained*, *wellconstrained*, or *underconstrained*. Well-constrained systems have a finite, albeit potentially very large number of *rigid* solutions; i.e., solutions that cannot be infinitesimally flexed to give another nearby solution: the solution space (modulo rigid body transformations such as rotations and translations) consists of isolated points - it is zero-dimensional. Underconstrained systems have infinitely many solutions; their solution space is not zero-dimensional. Overconstrained systems do not have a solution unless they are *consistently overconstrained*. In that case, they could be embedded within overall underconstrained systems, see Figure 1. Systems that are not underconstrained are called *rigid* systems.

A geometric constraint graph $G = (V, E, w)$ corresponding to geometric constraint system is a weighted graph with vertex set (representing geometric objects) V and edge set (representing constraints) E; $w(v)$ is the weight of vertex v and $w(e)$ is the weight of edge e, corresponding to the number of degrees of freedom available to an object represented by v and number of degrees of freedom (dofs) removed by a constraint represented by e respectively. See Figure 2.

A subgraph $A \subseteq G$ that satisfies

$$\sum_{e \in A} w(e) + D \geq \sum_{v \in A} w(v) \tag{1}$$

is called *dense*, where D is a dimension-dependent constant, to be described below. Function $d(A) = \sum_{e \in A} w(e) - \sum_{v \in A} w(v)$ is called *density* of a graph A.

The constant D is typically $\binom{d+1}{2}$ where d is the dimension. The constant D captures the degrees of freedom of a rigid body in d dimensions. For 2D contexts and Euclidean geometry, we expect $D = 3$ and for spatial

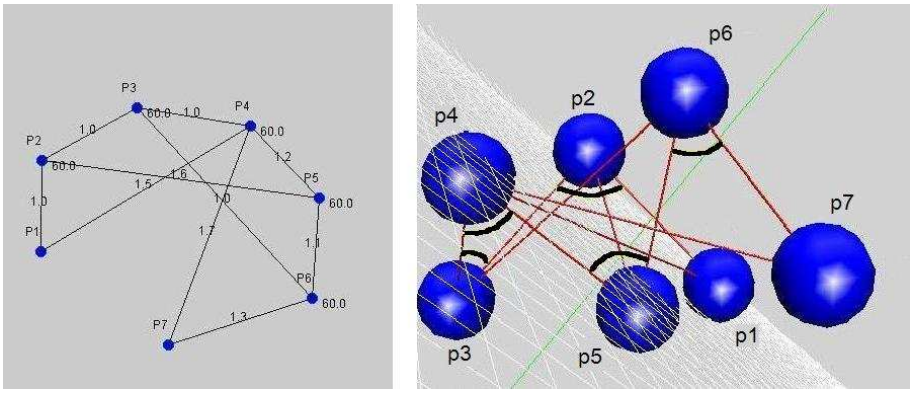

Figure 2: 3D constraint system drawn on 2D canvas with 7 point objects (p_1 - p_7) and 6 fixed length line segment objects (between point p_i and p_{i+1}); 4 distance constraints (equal distances); 12 incidence constraints where 2 linesegments are incident at each of the points p_1 -p_7; and 5 equal angle constraints between adjacent linesegments. The corresponding graph has 7 vertices of weight 3 (points) and 6 vertices of weight 6(fixed length linesegments); edges of weight 3 (an incidence constraint between 2 points remove the 3 degrees of freedom of one of the points) and edges of weight 1 (a distance constraint between 2 points remove 1 degree of freedom from one of the points; an angle constraint between 2 line segments removes 1 degree of freedom from one of the lines). Solution (right)

contexts $D = 6$, in general. If we expect the rigid body to be fixed with respect to a global coordinate system, then $D = 0$. A *trivial* subgraph is single vertex (in 2D) and a vertex or edge (in 3D).

Next we give purely combinatorial properties related to density that are used to detect generic algebraic properties. A dense, nontrivial graph with density strictly greater than $-D$ is called *dof-overconstrained*. A graph that is dense and all of whose subgraphs (including itself) have density at most $-D$ is called *dof-wellconstrained*. A graph G is called *dof-well-overconstrained* if it satisfies the following: G is dense, G has atleast one overconstrained subgraph, and has the property that on replacing all overconstrained subgraphs by dof-wellconstrained subgraphs (in any manner), G remains dense. Intuitively, this definition is used to prevent some overconstrained subgraphs with high density from skewing the classification of the entire graph. In particular, an extreme example could be a graph that has 2 subgraphs that are severely overconstrained, but with no constraints between them. By this definition, such a graph would not be well-overconstrained and would be correctly classified as underconstrained. A graph that is wellconstrained or well-overconstrained is called *dof-cluster*. A nontrivial dense graph is *minimal* if it has no nontrvial dense proper subgraph. All minimal dense subgraphs are dof-clusters but the converse is not the case. A graph that is not a dof-cluster is said to be *underconstrained*. If a dense graph is not minimal, it could in fact be an underconstrained graph: as pointed out, the density of the graph could be the result of embedding a subgraph of density greater than $-D$.

Next we discuss how the graph theoretic properties *degree of freedom (dof) analysis* relate to corresponding properties of the corresponding constraint *system*. For this, we need to introduce the notion of *genericity*. Informally, a constraint system is generically *rigid* if it is rigid (does not flex or has only finitely many non-congruent, isolated solutions) for most of choices of coefficients of the system. More formally we use the notion of genericity of e.g, [8]. A property is said to hold *generically* for polynomials $f_1, \ldots, f_n$ if there is a nonzero polynomial P in the coefficients of the f_i such that this property holds for all $f_1, \ldots, f_n$ for which P does not vanish.

Thus the constraint system E is generically rigid if there is a nonzero polynomial P in the coefficients of the equations of E - or the parameters of the constraint system - such that E is solvable when P does not vanish. For example, if E consists of distance constraints, the parameters are the distances. Even if E has no overt parameters, i.e, if E is made up of constraints such as incidences or tangencies or perpendicularity or parallelism, E in fact has hidden parameters capturing the extent of incidence, tangency, etc., which we consider to be the parameters of E. Examples are shown in Figure 3.

Laman's Theorem: [35]

A graph (V, E) is *rigid* in 2D *iff* $\forall E' \in E$ *s.t.*

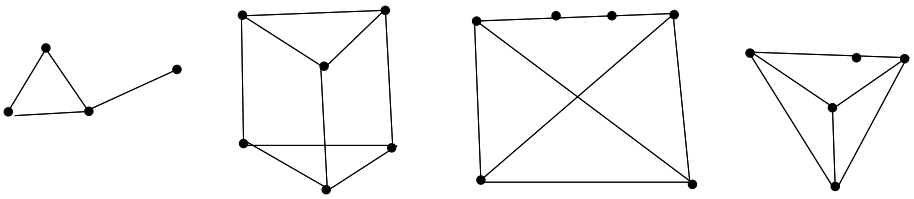

Figure 3: *(left)* not rigid, not generic rigid; *(middle left)* not rigid, generic rigid; *(middle right)* rigid, not generic rigid; *(right)* rigid, generic rigid

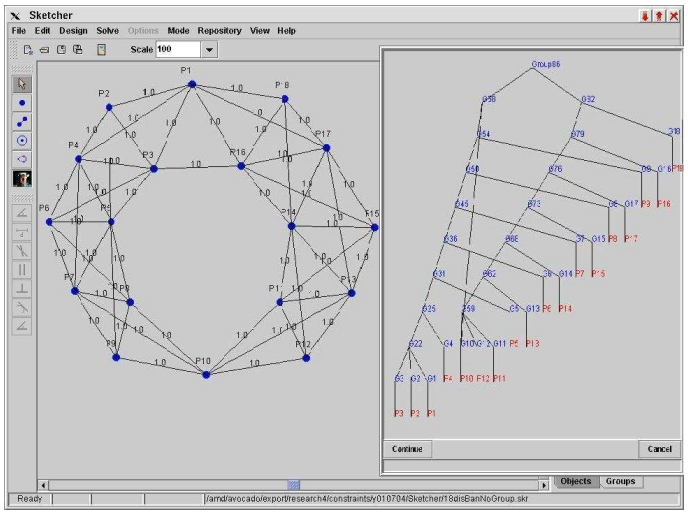

Figure 4: 3D constraint system with "bananas" type [16] generic constraint dependence not detectable by a simple dof count: the distance between p_1 and p_{10} is independently determined by the left and right half-moon clusters. It is however well-overconstrained and consistently constrained for the given set of distances: corresponding DR-plan has single root

1. $|E'| = 2|V'| - 3$;
2. $|F| \leq 2|V(F)| - 3$ for all non-empty subsets F of E'

In 2 dimensions, according to Laman's theorem, if all geometric objects are points and all constraints are distance constraints between these points then any minimal dense cluster represents a generically rigid system.

1.2 Bananas and Hinge Problem

In general, however, while generically rigid system always gives a cluster, the converse is not always the case. In fact, there are wellconstrained, dense clusters whose corresponding systems are not generically rigid and are in fact generically not rigid due to the presence of generic *constraint dependences*. See Figure 4 with 3D points and distance constraints, which illustrates the so-called "bananas" problem of [16], which generalizes to the so-called "hinge" problem [9, 10]. To date, there is no known, tractable, combinatorial characterization of generic rigidity of systems for 3 or higher dimensions even when only points and distances are involved [66], [16], although several conjectures exist. There are no known general combinatorial characterizations of 2D rigidity, when other constraints besides distances (such as angles) are involved. For constraint systems with angle and incidence constraints, but no distances such a characterization is given in [59]. For 3D points and distances, the notion of *module-rigid clusters* in [58] (an extension of dof-rigid clusters defined above) deals with all aspects of the bananas and hinge problems, i.e., it correctly characterizes generic rigidity in all known cases. Currently, no counterexamples are known - of module-rigid constraint graphs that are not generically rigid.

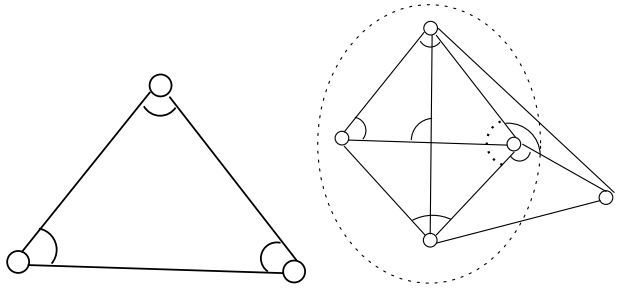

Figure 5: (*left*) explicit angle cycle; (*right*) implicit angle cycle

1.3 Angle Constraints System Conjecture

Point: point $p \in R^2$, is represented as (x, y).
Angle Constraint: the angle constraint between 2 ordered pairs of points, (P_{1i}, P_{2i}) and (P_{1j}, P_{2j}), satisfies that $\cos\theta = \frac{(P_{1i},P_{2i})\bullet(P_{1j},P_{2j})}{|(P_{1i},P_{2i})||(P_{1j},P_{2j})|} = \frac{(x_{2i}-x_{1i})(x_{2j}-x_{1j})+(y_{2i}-y_{1i})(y_{2j}-y_{1j})}{\sqrt{(x_{2i}-x_{1i})^2+(y_{2i}-y_{1i})^2}\sqrt{(x_{2j}-x_{1j})^2+(y_{2j}-y_{1j})^2}}$.
Geometric Angle Constraint System: a geometric system in which the objects are finite points and the constraints are finite angle constraints.
Explicit Angle Cycle: In one angle constraint system, if there is an angle cycle in its corresponding angle graph, we say this angle constraint system has an angle cycle. See Figure 5.
Implicit Angle Cycle: In one angle constraint system, for an angle cycle consists of assigned angles $\theta_1 \ldots \theta_s$ and unassigned angles $\gamma_1 \ldots \gamma_t$, $s, t \geq 0$, in its corresponding angle graph, if there exists $\theta_i \notin S_{\gamma_j}$ for $j = 1 \ldots t$, S_{γ_j} is the minimal 4 dof subgraph containts γ_j, we say this angle constraint system has an implicit angle cycle. See Figure 5.

Conjecture: (implicit) angle cycles are only sources of hidden dependences not detectable by standard dof analysis for a geometric angle constraint system.

1.4 Problems:

Here are the questions we mentioned in class. For G in 2D,

1. if underconstrained and not rigid, give all maximal rigid subgraphs
2. find minimum size of rigid subgraphs
3. if underconstrained and not over-underconstrained, a) give a complet set of edges *s.t.* anyone can be added that will not make it overconstrained. b) continue this process until it is wellconstrained [32]
4. if well-overconstrained a) give a complete set of edge *s.t.* anyone can be removed that will not make it underconstrained. b) continue this process until it is wellconstrained

2 Decomposition-Recombination Plan

Formally, a *dof-DR-plan* of a constraint graph G is a directed acyclic graph (dag) whose nodes represent dof-clusters in G, and edges represent containment. The leaves or sinks of the dag are all the vertices (primitive dof-clusters) of G. The roots or sources are a complete set of the maximal dof-clusters of G. For well or well-overconstrained graphs, the DR-plans have a single source. There could be many DR-plans for G. See Figures 6, 2, 4.

2.1 Overconstraints

First, each dof-cluster C in the DR-plan should be accompanied by a tractable representation of a complete list of *reducible* overconstraint sets directly associated with C. I.e., sets of constraints that do not lie entirely within

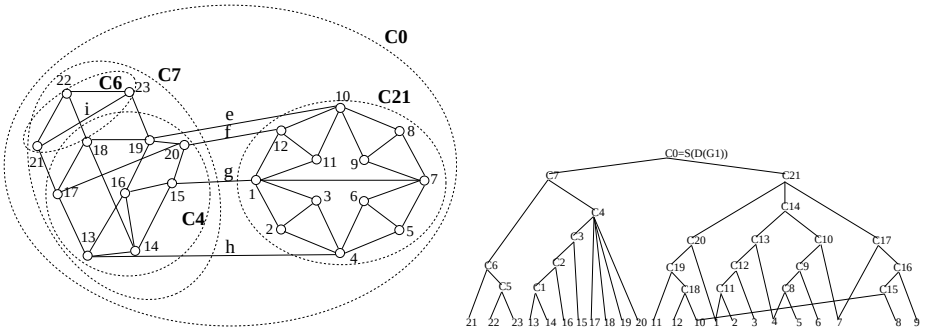

Figure 6: 2D constraint graph G1 and DR-plan; all vertices have weight 2 and edges weight 1

any child cluster of C and can be removed without affecting the dof-rigidity of C. The DR-planner should additionally admit an efficient method of removing overconstraints by making the appropriate changes to the DR-plan.

2.2 Optimality

The *size* of a cluster in a DR-plan is its fan-in or number of its children (it represents the size of the corresponding subsystem, once its children are solved). Since the algebraic-numeric solvers take time exponential in the size of the subsystems they solve, and the number of solutions is also typically exponential, minimizing the size of a DR-plan is *essential* to the ESM method presented here. An *optimal* DR-plan is one that minimizes the maximum fan-in.

It is shown in [38], [37], that the problem of finding the optimal DR-plan of even a 2D distance constraint graph is NP-hard, and approximability results are shown only in special cases. Nonapproximability results are not known.

One measure used in lieu of absolute optimality is based on the fact that most DR-planners make adhoc choices during computation (say the order in which vertices are considered) and we can ask how well (close to optimal) the *best* computation path of such a DR-planner would perform (on the worst case input). We call this the *best-choice approximation factor* of the DR-planner.

A more satisfactory measure of optimality is based on the following alternative property. A *tractable* DR-plan for systematic navigation should ensure that each cluster C should be accompanied by a a small set of its children C_i that form an *optimal covering set* of maximal clusters properly contained in C. A *covering set* of clusters is one whose union contains all geometric elements within C. The size of C is simply the size of this optimal covering set. The *optimality* here refers not to the size of the covering set, but to any suitable combinatorial measure of the algebraic complexity of the active constraint system for solving C, given the solutions of the child clusters in the covering set. This leads to the notion of *completeness* of DR-plans, given below.

2.3 Completeness

Any method that chooses an optimal covering set for a cluster C requires as input a generalized *complete decomposition of C into maximal proper subclusters,* formally defined as follows. The decomposition of any cluster C falls into one of 2 types. A Type 1 cluster C has exactly 2 child clusters, which intersect on a nontrivial subgraph, and their union covers all the geometric elements in C. A Type 2 cluster C has a set of child clusters C_i with the following property. The union of C_i's covers all the geometric elements in C; any pair of C_i's intersect on at most a trivial subgraph; and every C_i is a *proper maximal* subcluster of C, i.e., there is no proper subcluster of C that strictly contains C_i. Completeness is also needed for detecting implicit constraint dependences and for more accurate, module-rigid DR-planners.

2.4 Complexity

Another basic property of a DR-plan is its *width* i.e, *number* of clusters in the DR-plan to be small, preferably linear in the size of G: this reflects the complexity of the planning process and affects the complexity of the solving process that is based on the DR-plan. Clearly, this property competes with completeness.

Other desirable properties of DR-planners not mentioned above include systematic correction of underconstrained systems, and amenability to efficient updates of geometric primitives or constraints.

2.5 Problems

Here are the problems from Solidworks. (We showed the its software in class)

- Given S, get the solutions as close as possible to desired solutions
- Detecting the overconstrained system besides distance
- Knowing some operations are harder than others in solving and some constraints are more likely to be changed, how to solve the system to minimize the cost.
- Find the relationship between solution space and dof
- Change / rewriting the decomposition to minimize the minimal subsystem and preserve the set of solutions.
- a) Preoptimization of the system
  b) a good way to describe underconstrained solution space

3 Applications of Geometric Constraints Solving

3.1 Mechanical computer Aided Design

Solidworks' software is a commecial product using geometric constraints solving.

3.2 Enumeration of Self-Assembly Pathways for Symmetric macromolecular Structures

We consider the problem of explicitly enumerating and counting the assembly pathways by which an icosahedral viral shell forms from identical constituent protein monomers, see Figure 7. This poorly understood assembly process is a remarkable example of symmetric macromolecular self-assembly occuring in nature and possesses many features that are desirable while engineering self-assembly at the nanoscale.

The papers [55, 54] give the new model of that employs a static geometric constraint graph to represent the driving (weak) forces that cause a viral shell to assemble and hold it together, see Figure 8. The model was developed to answer focused questions about the structural properties of the most probable types of successful assembly pathways. Specifically, the model reduces the study of pathway types and their probabilities to the study of the orbits of the automorphism group of the underlying geometric constraint graph, acting on the set of pathways. Dr. Meera Sitharam gives a randomized algorithm to compute one measure of the probability of these pathways by faithfully sampling them.

The input to the algorithm is shown in Figure 9, the implementaion results are shown in Figures 10, 11

References

[1] Oswin Aichholzer, Gunter Rote, Bettina Speckmann, and Ileana Streinu. The zigzag path of a pseudotriangulation. In *Proc. 8th International Workshop on Algorithms and Data Structures (WADS) , Ottawa, Canada*, pages 377–388, 2003.

[2] S. Ait-Aoudia, R. Jegou, and D. Michelucci. Reduction of constraint systems. In *Compugraphics*, pages 83–92, 1993.

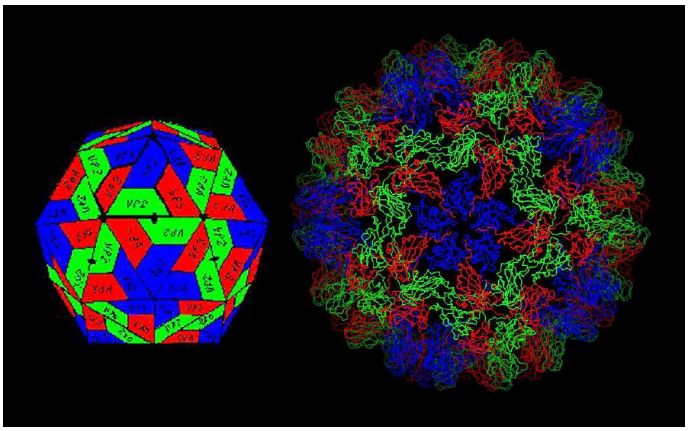

Figure 7: Virus Geometry

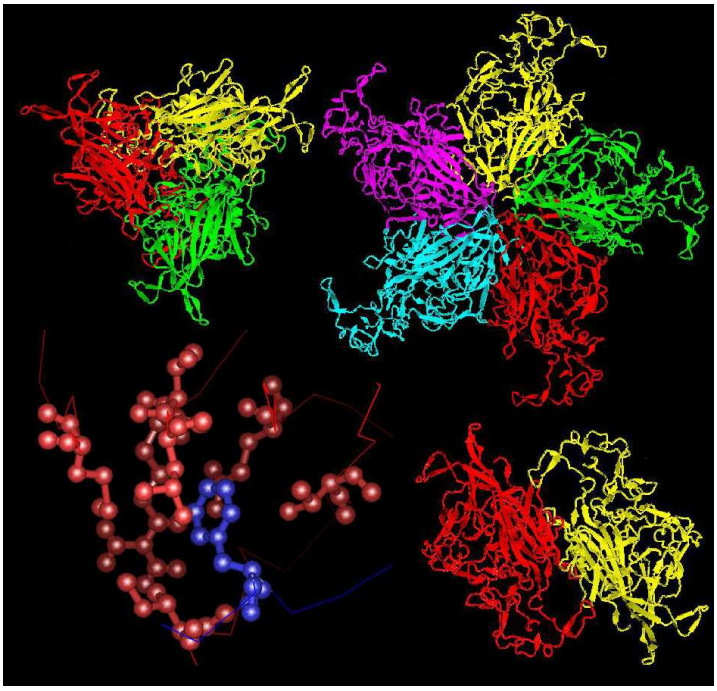

Figure 8: Atomic Interaction in Viruses: Pentamer, Dimer, Trimer

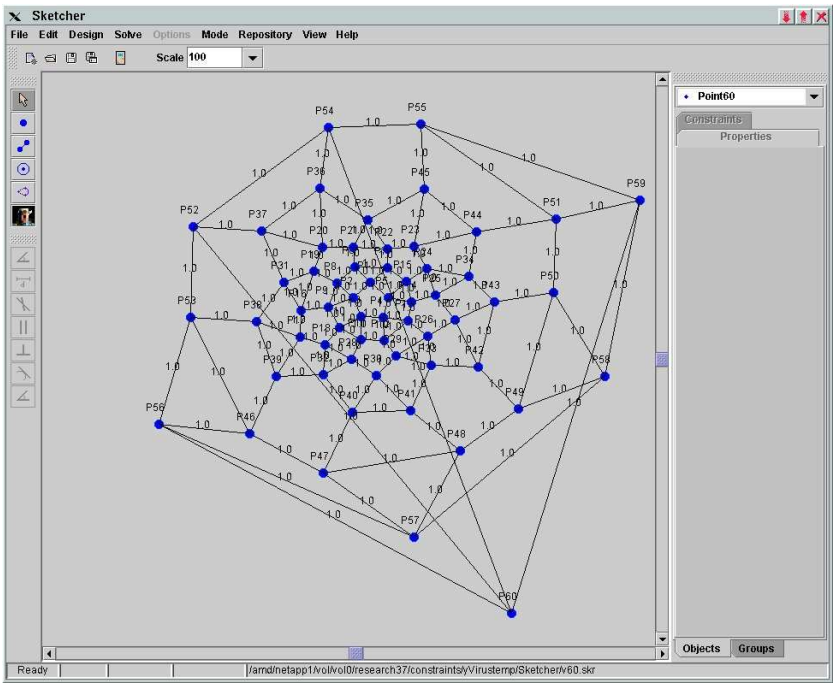

Figure 9: Input to the randomized algorithm

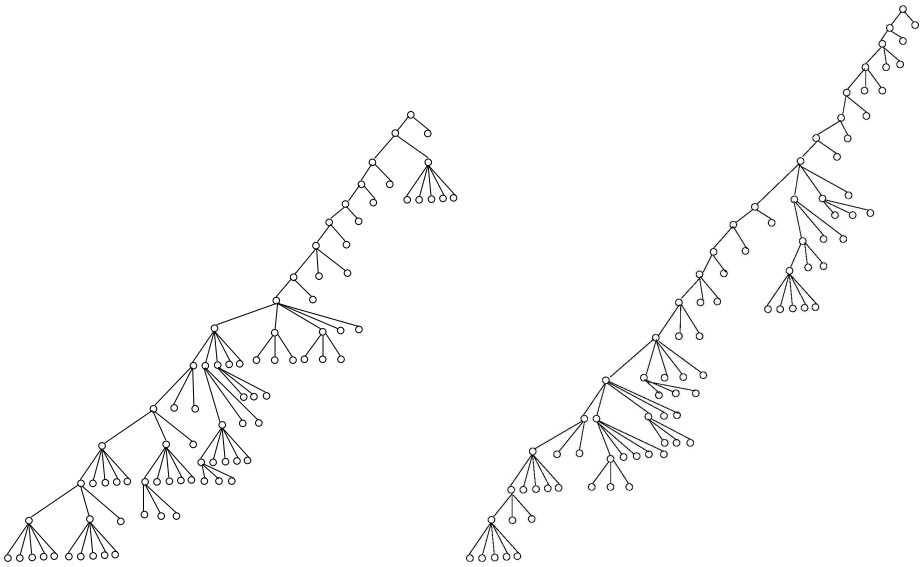

Figure 10: Implementation Results: 1) Highest and widest pathways in 2000 trials

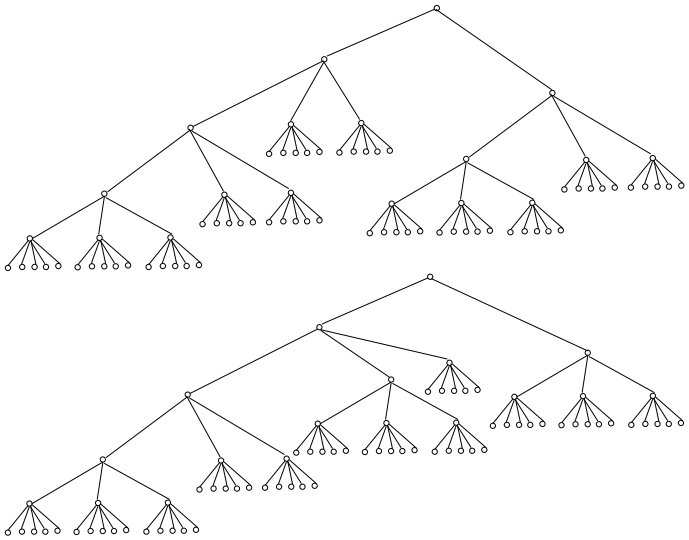

Figure 11: Implementation Results: 2) Two Isomorphism Classes

[3] László Babai and Wilfried Imrich. On groups of polyhedral graphs.. *Discrete Math.*, 5:101–103, 1973.

[4] László Babai and Wilfried Imrich. Sense preserving groups of polyhedral graphs. *Monatsh. Math.*, 79:1–2, 1975.

[5] Bernstein and David Naumovich. The number of roots of a system of equations. *Functional Analysis and its Applications (translated from Russian)*, 9(2):183–185, 1975.

[6] W. Bouma, I. Fudos, C. M. Hoffmann, J. Cai, and R. Paige. A geometric constraint solver. *CAD*, 27:487–501, 1995.

[7] B. Bruderlin. Constructing three-dimensional geometric object defined by constraints. In *ACM SIGGRAPH*. Chapel Hill, 1986.

[8] D. Cox, J. Little, and D. O'Shea. *Using algebraic geometry*. Springer-Verlag, 1998.

[9] Henry Crapo. Structural rigidity. *Structural Topology*, 1:26–45, 1979.

[10] Henry Crapo. The tetrahedral-octahedral truss. *Structural Topology*, 7:52–61, 1982.

[11] I. Fudos. *Constraint solveing for computer aided design*. Ph.D. thesis, Dept. of Computer Sciences, Purdue University, August 1995.

[12] I. Fudos. *Geometric Constraint Solving*. PhD thesis, Purdue University, Dept of Computer Science, 1995.

[13] Anna Gambin. On approximating the number of bases of exchange preserving matroids. In *Mathematical Foundations of Computer Science*, pages 332–342, 1999.

[14] X. S. Gao and S. C. Chou. Solving geometric constraint systems. I. a global propagation approach. *CAD*, 30:47–54, 1998.

[15] X. S. Gao and S. C. Chou. Solving geometric constraint systems. II. a symbolic approach and decision of rc-constructibility. *CAD*, 30:115–122, 1998.

[16] Jack E. Graver, Brigitte Servatius, and Herman Servatius. *Combinatorial Rigidity*. Graduate Studies in Math., AMS, 1993.

[17] Ruth Hass, David Orden, Gunter Rote, Francisco Santos, Brigitte Servatius, Herman Servatius, Diane Souvaine, Ileana Streinu, and Walter Whiteley. Planar minimally rigid graphs and pseudo-triangulations. In *Proc. ACM Symp. Comp. Geometry (SoCG) San Diego, California*, pages 154–163, 2003.

[18] C Hoffmann, M Sitharam, and B Yuan. Making constraint solvers more useable: the overconstraint problem. *to appear in CAD*, 2004.

[19] C. M. Hoffmann and C. S. Chiang. Variable-radius circles in cluster merging, Part I: translational clusters. *CAD*, 33:in press, 2001.

[20] C. M. Hoffmann and R. Joan-arinyo. Symbolic constraints in geometric constraint solving. *J. for Symbolic Computation*, 23:287–300, 1997.

[21] C. M. Hoffmann, A. Lomonosov, and M. Sitharam. Finding solvable subsets of constraint graphs. In *Constraint Programming '97 Lecture Notes in Computer Science 1330, G. Smolka Ed., Springer Verlag*, Linz, Austria, 1997.

[22] C. M. Hoffmann, A. Lomonosov, and M. Sitharam. Finding solvable subsets of constraint graphs. In Smolka G., editor, *Springer LNCS 1330*, pages 463–477, 1997.

[23] C. M. Hoffmann, A. Lomonosov, and M. Sitharam. Geometric constraint decomposition. In Bruderlin and Roller Ed.s, editors, *Geometric Constraint Solving*. Springer-Verlag, 1998.

[24] C. M. Hoffmann, A. Lomonosov, and M. Sitharam. Geometric constraint decomposition. In Bruderlin B. and Roller D., editors, *Geometric Constr Solving and Appl*, pages 170–195, 1998.

[25] C. M. Hoffmann, A. Lomonosov, and M. Sitharam. Planning geometric constraint decompositions via graph transformations. In *AGTIVE '99 (Graph Transformations with Industrial Relevance), Springer lecture notes, LNCS 1779, eds Nagl, Schurr, Munch*, pages 309–324, 1999.

[26] C. M. Hoffmann, A. Lomonosov, and M. Sitharam. Decomposition of geometric constraints systems, part i: performance measures. *Journal of Symbolic Computation*, 31(4), 2001.

[27] C. M. Hoffmann, A. Lomonosov, and M. Sitharam. Decomposition of geometric constraints systems, part ii: new algorithms. *Journal of Symbolic Computation*, 31(4), 2001.

[28] C. M. Hoffmann and J. Peters. Geometric constraint for CAGD. In M. Daehlen, T. Lyche, and Schumaker L., editors, *Mathematical Methods for Curves and Surfaces*, pages 237–254, 1995.

[29] C. M. Hoffmann and R. Vermeer. Geometric constraint solving in R^2 and R^3. In Du D. Z. and Hwang F., editors, *Computing in Euclidean Geometry*, pages 266–298, 1995.

[30] C. M. Hoffmann and B. Yuan. On spatial constraint solving approaches. In *Proc. ADG 2000, ETH Zurich*, page in press, 2000.

[31] B. Huber and B. Sturmfels. A polyhedral method for solving sparse polynomial system. *Math. Comp.*, 64:1541–1555, 1995.

[32] R. Joan-Arinyo, A. Soto-Riera, S. Vila-Marta, and J. Vilaplana-Pastó. Transforming an under-constrained geometric constraint problem into a well-constrained one. In *ACM Symposium on Solid Modeling and Applications,Proceedings of the eighth ACM symposium on Solid modeling and applications*, pages 33–44, 2003.

[33] G. Kramer. *Solving Geometric Constraint Systems*. MIT Press, 1992.

[34] G. Kramer. *Solving geometric constraint systems: a case study in kinematics*. MIT Press, 1992.

[35] G. Laman. On graphs and rigidity of plane skeletal structures. *J. Engrg. Math.*, 4:331–340, 1970.

[36] R. Latham and A. Middleditch. Connectivity analysis: a tool for processing geometric constraints. *Computer Aided Design*, 28:917–928, 1996.

[37] A. Lomonosov and M. Sitharam. Graph algorithms for geometric constraint solving. In *submitted, based on Lomonosov's Univ Florida PhD Thesis, 04*, 2004.

[38] Andrew Lomonosov. Graph and Combinatorial Analysis for Geometric Constraint Graphs. Technical report, Ph.D thesis, Univ. of Florida, Gainesville, Dept. of Computer a nd Information Science, Gainesville, FL, 32611-6120, USA, 2004.

[39] A. Middleditch and C. Reade. A kernel for geometric features. In *ACM/SIGGRAPH Symposium on Solid Modeling Foundations and CAD/CAM Applications*. ACM press, 1997.

[40] G. J. Nelson. A costraint-based graphics system. In *ACM SIGGRAPH*, pages 235–243, 1985.

[41] J. J. Oung, M. Sitharam, B. Moro, and A. Arbree. Frontier: fully enabling geometric constraints for feature based design and assembly. In *abstract in Proceedings of the ACM Solid Modeling conference*, 2001.

[42] J. Owen. www.d-cubed.co.uk/. In *D-cubed commercial geometric constraint solving software*.

[43] J. Owen. Algebraic solution for geometry from dimensional constraints. In *ACM Symp. Found. of Solid Modeling*, pages 397–407, Austin, Tex, 1991.

[44] J. Owen. Constraints on simple geometry in two and three dimensions. In *Third SIAM Conference on Geometric Design*. SIAM, November 1993. To appear in Int J of Computational Geometry and Applications.

[45] John C. Owen and Steve C. Power. The nonsolvability by radicals of generic 3-connected planar graphs. In *Automated Deduction in Geometry*, pages 124–131, 2002.

[46] John C. Owen and Steve C. Power. Are all 3-connected generic constraint configurations of points on a plane non-radical. In *Automated Deduction in Geometry*, 2004.

[47] J.A. Pabon. Modeling method for sorting dependencies among geometric entities. In *US States Patent 5,251,290*, Oct 1993.

[48] Giovanni Pistone, Eva Riccomagno, and Henry P. Wynn. *Algebraic Statistics: Computational Commutative Algebra in Statistics*. CRC Press, December 2000.

[49] J. Maurice Rojas. Why polyhedra matter in non-linear equation solving. In *Algebraic Geometry and Geometric Modelling (Vilnius, Lithuania, July 29-August 2, 2002), Contemporary Mathematics*, volume 334, pages 293–320, 2002.

[50] D. Roller. An approach to computer-aided parametric design. *CAD*, 23:303–324, 1991.

[51] M Sitharam. Frontier, an opensource 3d geometric constraint solver: algorithms and architecture. *monograph, in preparation*, 2004.

[52] M. Sitharam. Frontier, opensource gnu geometric constraint solver: Version 1 (2001) for general 2d systems; version 2 (2002) for 2d and some 3d systems; version 3 (2003) for general 2d and 3d systems. In *http://www.cise.ufl.edu/∼sitharam, http://www.gnu.org*, 2004.

[53] M Sitharam. Graph based geometric constraint solving: problems, progress and directions. In D. Dutta, R. Janardhan, and M. Smid, editors, *To appear in AMS-DIMACS volume on Computer Aided Design*, 2004.

[54] M. Sitharam and M. Agbandje-Mckenna. Modeling virus assembly pathways using computational algebra and geometry. In *Proceedings of the 10th Applications of Computer Algebra conference*, 2004.

[55] M Sitharam and M Agbandje-Mckenna. Sampling virus assembly pathway: Avoiding dynamics. *accepted to Journal of Computational Biology, available upon request*, 2004.

[56] M Sitharam, A Arbree, Y Zhou, and N Kohareswaran. Solution management and navigation for 3d geometric constraint systems. *accepted to ACM TOG, available upon request*, 2004.

[57] M Sitharam, J Peters, and Y Zhou. Solving minimal, wellconstrained, 3d geometric constraint systems: combinatorial optimization of algebraic complexity. *Automated deduction in Geometry (ADG) 2004, available upon request*, 2004.

[58] M Sitharam and Y Zhou. A tractable, approximate, combinatorial 3d rigidity characterization. *Fifth Automated Deduction in Geometry (ADG)*, 2004.

[59] M. Sitharam and Y. Zhou. Characterization of rigidity for 2d angle and incidence constraints. *Manuscript; available upon request*, 2005.

[60] A. Sommese and J. Verschelde. Numerical homotopies to compute generic points on positive dimensional algebraic sets. *Journal of Complexity*, 16(3):572–602, 1999.

[61] N. Sridhar, R. Agrawal, and G. L. Kinzel. An active occurrence-matrix-based approach to design decomposition. *CAD*, 25:500–512, 1993.

[62] Ileana Streinu. A combinatorial approach to planar non-colliding robot arm motion planning. In *Proc. 41st ACM Annual Symposium on Foundations of Computer Science (FOCS)*, pages 443–453, 2000.

[63] Ileana Streinu and Walter Whiteley. Single-vertex origami and 3-dimensional expansive motion. submitted,Dec. 2003.

[64] P. Todd. A k-tree generalization that characterizes consistency of dimensioned engineering drawings. *SIAM J. Discrete Mathematics*, 2:255–261, 1989.

[65] D. C. Gossard V. Lin and R. Light. Variational geometry in computer-aided design. In *ACM SIGGRAPH*, pages 171–179, 1981.

[66] W. Whiteley. Rigidity and scene analysis. In *Handbook of Discrete and Computational Geometry*, pages 893 –916. CRC Press, 1997.