Generating 3D Spatial Layouts using A-Teams

S. Sachdev, C.J. Paredis, S.K. Gupta, S.N. Talukdar
Carnegie Mellon University, Pittsburgh, PA 15213
corresponding author: Sanjay Sachdev
address: EDRC/ICES, Carnegie Mellon University, Pittsburgh, PA 15213.
telephone: (412) 268-2257
email: sachdev@cs.cmu.edu

ABSTRACT

Spatial layout is the problem of arranging a set of components in an enclosure such that a set of objectives or constraints is satisfied. The constraints may include non-interference of objects, accessibility requirements or connecting cost limits. Spatial layout problems are found primarily in the domains of electrical engineering and mechanical engineering in the design of integrated circuits and mechanical or electromechanical artifacts. Traditional approaches include heuristics, Genetic Algorithms and Simulated Annealing. The A-Teams approach provides a way of synergistically combining these approaches in a modular agent based fashion. A-Teams are also open to the addition of new agents. Modifications in the task requirements translate to modifications in the agent mix. In this paper we describe how A-Team based optimization can be used to solve 3D spatial layout problems.

Keywords: Asynchronous Teams, Software Agents, Spatial Layout, Electro-mechanical Design, Optimization.

1. INTRODUCTION

A number of design and manufacturing problems require spatial arrangements of parts subject to a wide variety of spatial constraints. Given a set of arbitrarily shaped rigid 3 dimensional solid objects, an arbitrarily shaped housing (which may be optional in some cases), the goal is to arrange the objects spatially inside the housing without violating any spatial constraint and optimize the given objective function(s). Example problems include the design of satellites, the design of cars, packing for material handling, and 3D nesting for rapid prototyping. We collectively refer to such problems as spatial layout problems. Different spatial layout problems have different types of constraints and objective functions. Most spatial layout problems have many different conflicting constraint and objective functions which makes them very challenging. It is very difficult for humans to efficiently solve large problems involving a large number of complex parts. The specialized software currently used for solving such problems is difficult and expensive to develop.

There has been considerable research in 2D spatial layout problems found in integrated circuit design. Extensions of such problems to three dimensions consist of packing 2D layers. The objects are almost always rectangular in shape. Problems in material handling packing are characterized by packing box shaped containers. Most of the research in the operational research, industrial engineering and electrical engineering communities has been on packing such box-like objects, usually into box-like containers. Mechanical engineering problems on the other hand are usually 3-D in nature. However there has not been much work in this area due to the lack of computational resources. It is only recently that computers have become powerful enough to attempt automated optimization of 3D spatial layouts. Recently developed technologies such as rapid prototyping [Hinzman95] also create a need for better 3D spatial layout tools.

Spatial layout problems in 3D are usually solved using heuristics, genetic algorithms (GA) or simulated annealing (SA). The heuristics are designed for packing rectangular boxes inside other rectangular boxes. GA and SA methods have been developed that apply to non-convex, irregular objects, but most applications and demonstrations have been for fairly simple, regular objects (such as cuboids and cylinders). While GA and SA are global optimization methods, both require many constraint and objective evaluations in the search for good solutions. Additionally, both methods attempt to optimize with respect to a single merit function, expressed as the weighted sum of all objectives and constraints. They are both sensitive to the choice of weights.

We are developing a different approach to solving spatial layout problems. Our approach automatically determines optimal spatial layouts of the physical components while satisfying constraints on component interference, accessibility, connectivity, and separation. The individual constraints are captured in software agents that interact with each other in an autonomous and asynchronous manner through shared memories containing candidate solutions. The agents are implemented in a distributed fashion and can be reconfigured dynamically, for instance, to capture changes in the design constraints and requirements. To allow the designer to weigh the trade-offs in the constraints and optimization criteria, the Pareto optimal frontier of non-dominated solutions is computed and visualized. Our approach has the following advantages: (1) it has core technology components that can be used; (2) additional constraints and objective functions can be easily added; (3) agents can be combined together in variety of ways; (4) humans can participate in the problem solving process; and (5) the system is based on a distributed architecture which can use all the available computational power

2. SPATIAL LAYOUT

Spatial layout problems in Mechanical Engineering domains such as design, prototyping and manufacturing typically require packing components with complicated shapes, subject to a variety of constraints. We have developed a tool to solve spatial layout problems with an arbitrary number and variety of components and constraints. We selected a small set of representative constraints in order to develop and demonstrate the spatial layout tool. Given a housing and a set of components {Component_i}, we specify the following constraints:

c1. **Protrusion**: $\text{Volume}(\text{Component}_i \cap \text{HousingPackableVolume}) = \text{Volume}(\text{Component}_i), \forall\ i$

c2. **Overlap**: $\text{Volume}(\text{Component}_i \cap \text{Component}_j) = 0, \forall\ i, j; i \neq j$

c3. **Connectivity**: minimize $\Sigma_{i,j}\ (W_{ij} * \text{Distance}(\text{Component}_i, \text{Component}_j))$, W_{ij} is the cost of connecting the two components

c4. **Separation**: $\text{Distance}(\text{Component}_i, \text{Component}_j) >= \text{MinSeparation}_{ij}$

c5. **Accessibility**: $\text{Volume}(\text{SweepFace}_{ki}(\text{Component}_i) \cap \text{Component}_j) = 0, \forall\ i, ki \text{ and } j; i \neq j$

c6. minimize $\Sigma_{i,ki}\ (\text{Volume}(\text{SweepFace}_{ki}(\text{Component}_i) \cap \text{HousingPackingVolume}))$

where:

HousingPackableVolume = the space inside the housing enclosure into which the components must be placed

$\text{Volume}(\text{Component}_i)$ = volume of Component_i

$\text{Distance}(\text{Component}_i, \text{Component}_j)$ = a distance measure proportional to the cost of connecting the two components

$\text{SweepFace}_{ki}(\text{Component}_i)$ = an object created by sweeping face k on Component_i in the direction of its normal, to infinity.

The connectivity constraint (c3) measures the total cost of establishing physical connections between components. Separation constraints (c4) keep components apart that might interfere with each other electromagnetically or thermally. Certain faces on some of the components are required to be accessible (i.e. not blocked by any other component, for instance the battery compartment of a portable radio needs to be accessible). Accessibility (c5) of a face is defined as the ability to sweep a face along the direction of its normal without the swept volume intersecting with any other object. An alternate definition of accessibility (including both c5 and c6) was also used to demonstrate the reconfigurability of the A-Team based tool. The alternate problem definition measures accessibility not only by the ability to sweep the required face freely, but also by the distance of the face from the boundary of the housing measured by the intersection between the swept face volume and the housing.

Constraint (c3) is designed to represent differentiable and continuous constraints. Constraint (c4) represents discontinuous constraints with many regions of zero violation. Constraint (c5) represents procedurally defined constraints. Both constraints (c4) and (c5) have highly discontinuous derivatives, which makes them difficult to solve using gradient based methods.

2.1. Spatial Layout: Problems and Methods

We will now relate this class of problems to problems and approaches considered in the literature. There are three primary areas of research in spatial layout problems. These are Operations Research (OR) and Industrial Engineering (IE) where the problems are usually referred to as "packing" problems; Electrical Engineering (EE), where the problems are called "layout" or "placement" problems; and Mechanical Engineering (ME), where the problems are called "configuration design" problems.

The focus in EE emphasizes connectivity costs between objects since there is some freedom on the size and shape of objects (which may lie within some neighborhood of the given target). The problems derive from the need to transform a circuit description, consisting of a set of modules and nets (wiring connecting the modules), into a layout in 2D which is then implemented in silicon to make integrated circuits. There are several steps to this process, consisting of: partitioning the circuit into components; placement of modules of fixed sizes without overlap onto a 2D surface representing the chip so as to minimize wiring requirements; routing the wiring around the modules on the surface; and compaction of the final arrangement. Placement usually implies locating rectangles of fixed size. Floorplanning is placement of rectangles that may vary in size. The main concern is to fit the rectangles and minimize the length (and quality, as measured for example by the number of bends) of wiring used to connect components. A good current survey for EE problems is found in [ALLP97]. Problems in EE are primarily 2D due to the planar nature of the integrated circuit technology. There is some extension into the third dimension, but this consists of stacking layers of components, which can be considered as a set of 2D problems. The objects and container are rectangular in shape. Such problems consider much simpler objects than the irregular, non-convex, 3D objects found in electromechanical assemblies.

The focus in OR, IE and ME has been mainly on maximizing packing density since the components in mechanical assembly and container packing are rigid. Other constraints such as stability or accessibility are usually ignored. Much of the work in OR is on 2D problems related to theoretically oriented bin-packing or knapsack problems. This work focuses on packing rectangles in rectangular boundaries. The OR literature focuses on the development of heuristics (for 2D or 3D box-like objects in box-like housings). Nonlinear programming methods are not used due to the extreme difficulty in problem formulation and the disconnected character of feasible regions in spatial layout problems. However there has been some exploratory investigation into other methods such as SA or Tabu Search, but there seems to be little interest. A general survey of packing problems in OR is [DD92]. It describes the main types of problems that have been considered in the literature, classifying the problems by dimension (1, 2 or 3D). This paper also describes the use of non-heuristic based methods (such as branch and bound) for 2D rectangle packing, which require considerable computation time even for small problems. The literature focuses on packing density, usually ignoring other (possibly important) constraints such as accessibility or separation requirements. The 3D problems in OR or IE usually involve packing cuboids into other cuboids (e.g. for packing boxes in containers for shipping). In addition to the packing constraints, the additional constraint of locating the center of gravity of the packed container low and near the center (in the horizontal plane) of the container is often considered. We consider a larger variety of constraints in addition to dealing with more complex objects.

2.2. Solution Techniques described in the Literature

Current work in the packing of rigid objects with a focus on practical problems uses heuristic based algorithms or randomized methods. There are three main categories of solution methods described in the literature: Heuristics, Genetic Algorithms, and Simulated Annealing. Gradient based optimization methods such as nonlinear programming methods are normally not applied to layout problems because of the difficulty in setting up the required problem formulation.

This section provides a brief description and discussion of the popular methods.

Heuristics: These are most widely used in the OR and IE literature. For three dimensional problems, [DD92] states that the approaches are "entirely composed of a series of ad-hoc heuristic rules derived from common sense" but that "practical experience suggests that any of [these] methods will out-perform manual methods on average." The section on non-rectangular packing indicates the absence of methods here, with most of the work being in circle and sphere packing, primarily in the mathematical and recreational literature. It is pointed out that existing heuristics designed for rectangles are difficult to extend to these problems. Most heuristics locate the objects one at a time in the container, with some kind of layering rule to indicate how objects will be placed. The search space is usually limited to placing objects so that their corners align with already placed objects. Examples of such heuristics can be found in [Lengauer90, Aziz91, GMM90, HT90].

Genetic Algorithms (GA): are modeled on the principle of natural selection [Davis91]. Solutions are represented as binary strings (chromosomes). Candidate solutions grouped in populations are maintained. The populations are ranked based on a "fitness function," which is usually the merit function being optimized. New solutions are generated by combining two or more solutions (crossover) and by perturbing solutions (mutation). This process, called reproduction, is modeled on natural selection. Solutions with better fitness have a better chance of being selected for reproduction. Genetic algorithms might either store the solution directly, or might make use of an encoded representation along with a decoder which maps from the solution string to an actual solution. For example, in packing problems, the solutions might consist of the location and orientation of the objects (direct representation), or might be an ordering of the objects (encoded representation) which is then mapped into an actual location. Alternatively, a combination of the two representations may be used such as an ordering and orientation for the objects, where the encoded part is used only to determine translations. Genetic Algorithm approaches to packing [KMK91, WF94, IBKRW97], typically use partially or completely encoded representations with a packing heuristic to locate the parts. They extend the heuristic methods by allowing for some randomization. The use of packing heuristics greatly limits the size of the search space - and ensures feasibility in terms of overlap and protrusion. GA in 3D have been applied to packing cuboids in cuboids while optimizing the location of the center of gravity in the X-Y plane [KMK91, WF94]. Extensions to cuboids with holes (shoebox-like objects with thick walls) have recently been tried [IBKRW97] but the size of the problems has been very small (2 objects with holes, plus 3 cuboids) and no objectives have been considered.

Simulated Annealing (SA): is a method of optimization based on the physical process of annealing metals [KGC83]. SA is an iterative method. Solutions are subject to random perturbations. Perturbations that lead to better merit function values are always accepted. Perturbations that lead to worse values are sometimes accepted based on a time dependent "temperature" parameter and the amount of change in the merit function. If a perturbation is accepted the current solution is replaced by the perturbed solution. The proportion of worse perturbations accepted decreases with time and the algorithm evolves in a continuous fashion from behaving like a random walk to behaving like a hill climb. SA has been used extensively for layout problems in integrated circuit design. This led to the use of SA for packing 3D objects [SC93, SC95], which demonstrated that SA works well for packing cuboids and cylinders in cuboids. [Szykman95] also shows examples of using SA for packing cylinders subject to connectivity costs between objects.

Other Methods: An atypical approach to packing can be found in [KG91] where a nonlinear programming type approach is used. The formulation of the problem is extremely complicated (the objects are represented as unions and intersection of half spaces, which grows very rapidly with the complexity of objects) and the only example with 3D objects has only 3 objects, one of which is a cuboid. It appears unlikely that such an approach will ever be competitive with heuristic or randomization based methods. Another approach which has been considered is Tabu Search. For example [Dowsland93] states that it would probably work better than SA. There is little experimentation to confirm or deny this claim.

In contrast to the research described in the literature and summarized above, we are considering a much more general class of problems than covered in the literature: 3D spatial layout problems that are larger in terms of the complexity of objects considered and in the number and complexity of constraints. This richer class of problems has not been effectively tackled. Heuristic methods are fast, but are fragile and difficult to extend as the number and characteristics of constraints change. These methods work best with cuboids. General purpose randomized methods such as SA and GA are broadly applicable but require many evaluations, restricting applications to fairly simple objects (cuboids and cylinders). The A-Teams approach, which we will describe next, allows us to combine heuristics and randomized methods in one unified framework. It has the advantage that one can easily add new solution algorithms to accommodate variations in the problem definition. As a result, the A-Teams approach is a perfect match for solving 3D layout problems for which the objectives and constraints vary from one application to another. Moreover, by combining heuristics with randomized solution methods, A-Teams provides a good trade-off between solution speed and robustness.

2.3. Asynchronous Teams of Autonomous Agents (A-Teams)

An A-Team is an organization of autonomous (cyber) agents (or agent, for short). An agent is an entity whose input and output spaces are maintained by computers, and whose control system is completely self-contained (i.e. it decides for itself when and what to work on). Agents collaborate by exchanging data via shared repositories of data called memories. Agents consist of three components: a scheduler to determine when the agent will work; a selector that chooses what the agent will work on; and an operator that modifies solutions and writes data to the output space. There are two classes of agents in A-Teams: constructors and destroyers. Constructors add new solutions or modify solutions in their output memory. Destroyers prevent overflow of the memories by erasing undesirable solutions (e.g. solutions with bad evaluations). Constructors can be further categorized as creators (that generate new solutions to seed the memories), modifiers (that improve upon existing solutions) and evaluators (to evaluate solutions in the memories). The organization of agents and memories can be represented as directed hypergraphs, where nodes represent memories and arcs represent agents. Such a graph is called a dataflow. A dataflow is an A-Team if and only if all arcs lie in closed loops. A complete set of definitions, descriptions and prescriptions for A-Teams can be found in [Talukdar98]. Additional details and theoretical analyses of why A-Teams work well can be found in [TBGD97].

The power of A-Teams lies in their modular design. Solutions in various representations (or even solutions to related problems that might provide hints at solutions for the problem at hand) are stored in different memories. Heuristics that operate on these solutions are encapsulated as autonomous agents (they work independently and asynchronously of each other). There is no central controller as in hierarchical problem solvers. Thus agents can be added or removed as needed without affecting existing agents in the team. Each agent is independent of all others, and there may even be multiple copies of agents running simultaneously in parallel. Furthermore, the A-Team can be easily scaled as more processors become available, and the A-Team configuration may even be changed dynamically by adding and removing agents.

A-Teams have been developed for a variety of computation problems, including nonlinear equation solving [TPG83], traveling salesman problems [deSouza93], high rise building design[Quadrel91], reconfigurable robot design [Murthy92], diagnosis of faults in electric networks [Chen92], control of electric networks [TR93], job shop scheduling [CTS93], steel [LMHM95] and paper mill scheduling [RWMTSAFAYHJ96], train scheduling [Tsen95], protein folding [LGTH97] and constraint satisfaction [GHSTM]. Not only do these asynchronous teams find good solutions, but they appear to be scale effective: solution quality and speed improve with the addition of agents and computers.

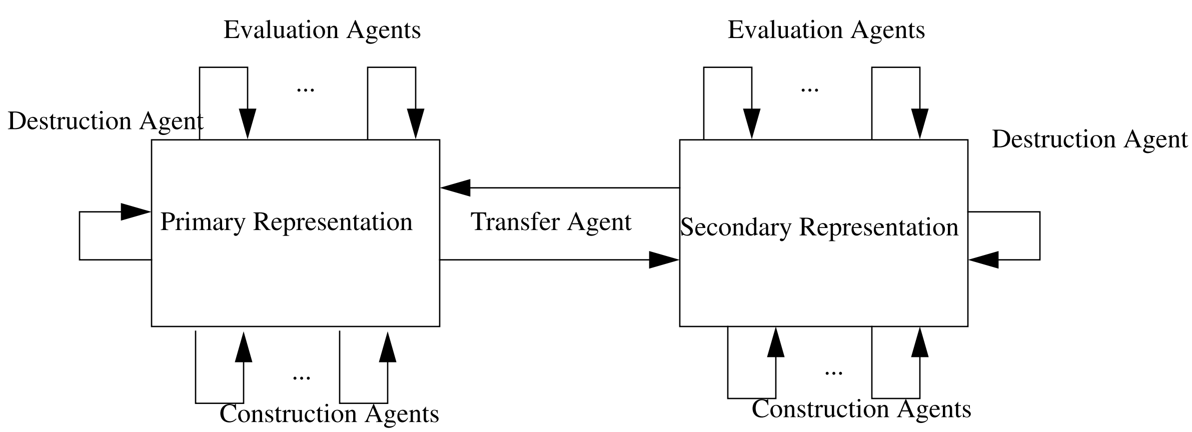


Figure 1. An A-Team

3. SPATIAL LAYOUT A-TEAM

The modularity of A-Teams allows them to be quickly and effortlessly reconfigured to match the characteristics of a particular problem instance. This characteristic makes A-Teams well suited for solving 3D spatial layout problems. The agents for improving solutions or evaluating solutions can all work in parallel across a network of computers. No coordination is required, facilitating the assembly of the team. Also, since A-Teams work with populations of solutions and do not require a single merit function, sets of non-dominated solutions (a Pareto frontier) can be constructed. As a result, the user does not have to provide a set of weights a-priori and can choose the best trade-offs between the different objectives after good solutions have been found.

The A-Team based spatial layout tool is designed to solve problems with multiple constraints, without restrictions on the number or types of constraints. Modularity was considered imperative and separate agents were developed for each constraint. We refer to agents that work only on one or a small number of constraints as specialist agents. Specialist agents are experts on their own constraints and ignorant of others. The ability of specialist agents, each of which has knowledge about only one constraint (or objective), to solve problems with multiple constraints has been demonstrated in the domain of nonlinear programming in [TSC97]. The advantage of partitioning problems in this manner is that the component agents are more likely to be reusable. Whenever a new type of problem needs to be solved, the appropriate set of specialist agents can be assembled from a repository. Customization of solvers translates to the selection of an appropriate mix of agents.

We use a typical electromechanical assembly design problem as an instance of the class of problems under consideration. The task consists of packing a set of components into a housing unit subject to a set of constraints. There are accessibility constraints (for components such as knobs, connectors, heat sinks etc.); connectivity requirements (for components that may have many connections between them); and separation constraints (for components that may interfere electromagnetically or thermally if placed too close together).

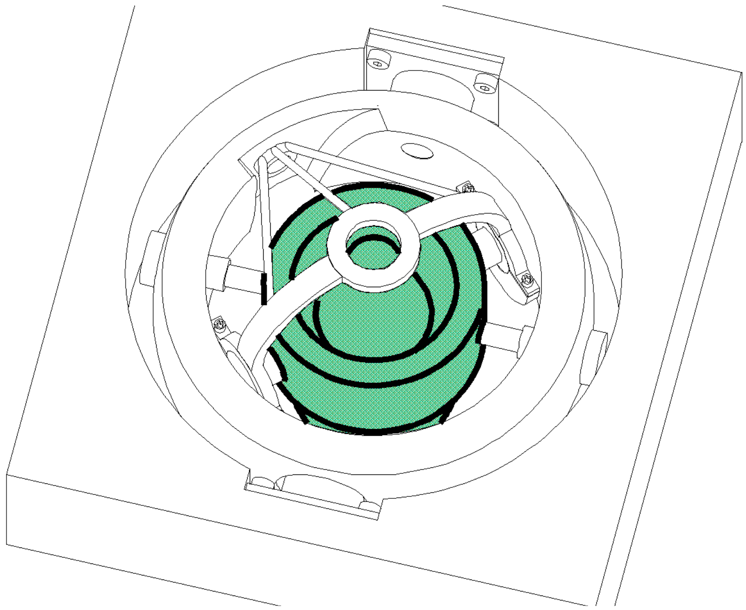


Figure 2. The Seeker Assembly for a Missile (The optics housing is colored)

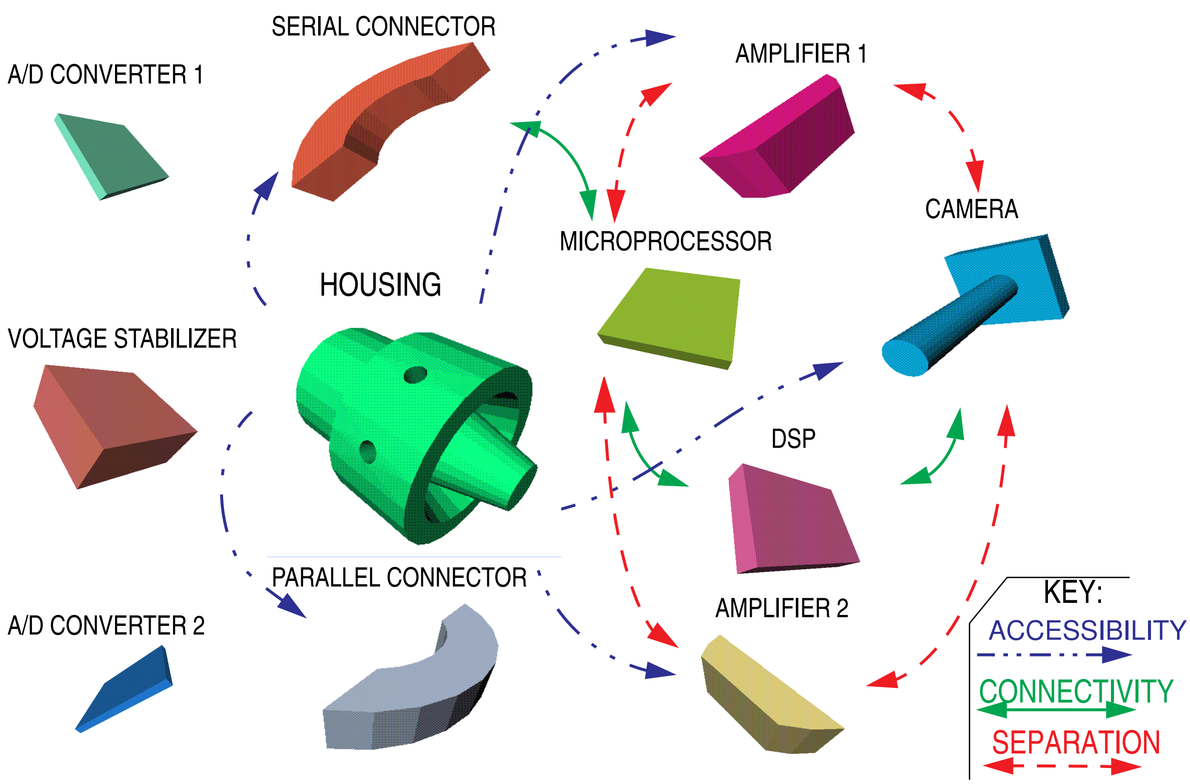


Figure 3. Problem Description: Optics Housing, Components and Constraints

The artifact is the optics subassembly of a missile seeker shown in Figure 2. There are ten components in the subassembly: a ccd camera unit, a microprocessor, a digital signal processor, two analog-digital converters, a voltage stabilizer, serial and parallel connectors and two amplifiers. There are connectivity requirements between the microprocessor, digital signal processor, camera and connectors. The amplifiers must be kept suitably far from the camera and microprocessor. Additionally, the connectors must be accessible from outside the housing, and there must be no component blocking the front of the camera lens. The housing is non convex, and contains holes. The components cover a range of shapes and sizes. The shapes of the objects and constraints between them are illustrated in Figure 3.

The objects are represented by 3D CAD models in ACIS format. The ACIS 2.1 class library and API are used for manipulation and processing of the 3D models. Solutions are represented as vectors of translations of the objects from their reference locations in the world coordinate system. A secondary representation uses a discretized grid, with solutions represented as a list of grid points corresponding to the centroid location of each object. ACIS 2.1 is used to evaluate the interference, protrusion, and accessibility requirements. The agent for viewing solutions uses OpenInventor for display and rendering.

A repository consisting of two types of memories (corresponding to the two representations used) and a collection of agents was developed. Such agents can be easily assembled into A-Teams resembling the example dataflow provided in Figure 1. This is demonstrated by modifying the specifications of the example problem. Because the A-Team was designed to be modular (separate independent agents for each constraint), redefining the accessibility metric required only that the accessibility agents be replaced while the remainder of the team continued operation.

3.1. Agents

There is an independent **evaluation agent** for each constraint. Evaluation agents start work whenever they detect new unevaluated solutions in the memory. Since they are independent, they can simultaneously update evaluations for new solutions or even work on different solutions.

Agents which generate new solutions are called **construction agents**. These are classified into two categories: modification agents which use existing solutions as starting points to create better solutions, and creation agents which generate new solutions to seed the memories.

There are independent **modification agents** for each constraint. Since the protrusion and interference constraints are difficult to repair, agents for accessibility, separation and connectivity attempt not to increase violations of these constraints. These agents use a variety of heuristics, restricted only by the availability of good methods. As an example, one heuristic for connectivity (the one used in the A-Team) uses the gradient of the connectivity cost to improve solutions. Because the modular nature of the team allows multiple agents for a constraint to run concurrently, other agents using different algorithms to improve connectivity may also work in parallel.

The current set of **creation agents** can be classified into three groups: one that uses an ordering of the objects combined with a decoder (similar to packing heuristics found in the Operations Research literature and in Genetic Algorithms), one that uses random placement of the objects (similar to that used in Simulated Annealing or Tabu Search) and one that uses a discretized grid to locate objects. We will describe one of the creation agents as an example of how secondary representations can be used to assist in optimization. An A-Team that uses a discretized grid representation ignores protrusion and interference, trying to place accessible objects near the grid boundary. Separation and connectivity are approximated by integral manhattan distances on the grid. The only goal of this subsidiary A-Team is to generate good starting configurations.

A **destruction agent** erases solutions that are dominated by others in the memory. If the memory is full of non-dominated solutions, an arbitrarily selected solution may be destroyed to make space.

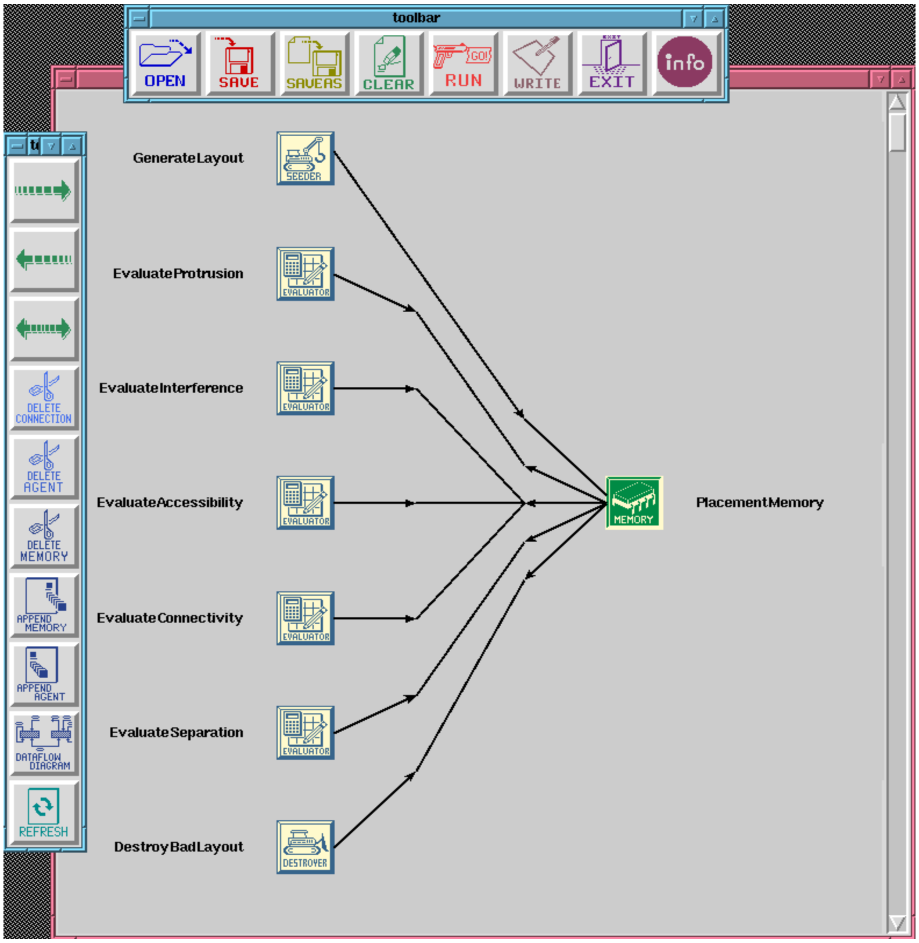
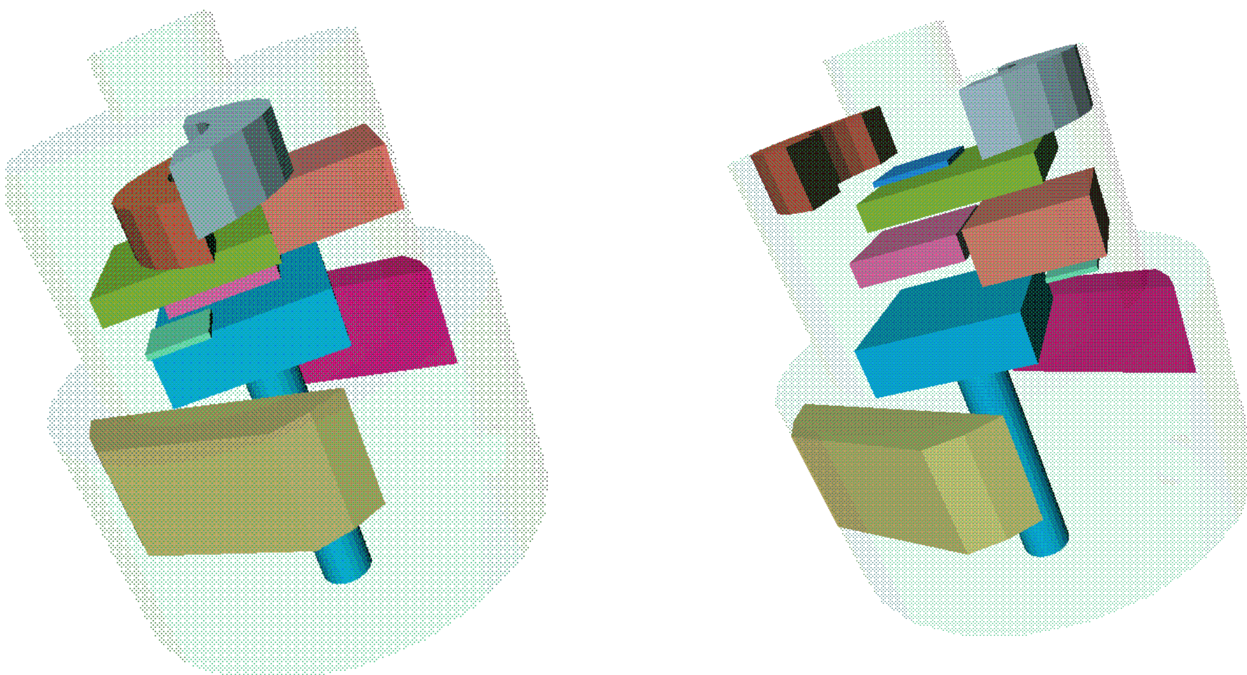


Figure 4. The A-Team Toolkit

3.2. Tests

The tool was first used to find a placement of the objects given the accessibility, connectivity, separation, interfere and protrusion constraints. A solution satisfying the constraints is shown in the Figure 5a. The task was then modified by redefining the measure of accessibility. It was considered insufficient for the appropriate face of the connectors to just be reachable from outside, and accessibility was redefined to give preference to placements where the connectors and the heat sinks of the amplifiers were closer to the housing boundary. In order to keep the system modular and to simplify the development of agents, there is a separate agent designed to improve each constraint. Keeping a clear distinction between the sub-tasks assigned to each agents makes it easy to reconfigure the team should task requirements change. We have shown this by modifying the specifications for accessibility, and by reconfiguring the A-Team just by replacing the accessibility related agents. The remaining agents were unchanged and, in fact, continued working while the accessibility related agents were reconfigured. A solution found by the modified team is shown for comparison in Figure 5b. Some of the components have been moved closer to the housing. Neither of the two solutions dominates - the first is better in terms of connectivity, while the second is better for accessibility. The final choice is left to the user.



5a. Layout 1

5b. Layout 2

Figure 5. Alternate Layouts for the Optics Assembly

3.3. Pareto Frontiers

The functionality of the team was extended to construct and display Pareto frontiers by adding a new agent. This agent retrieves solutions from the memory and provides plots of the objectives, allowing the user to determine solutions that are on the Pareto frontier. Examples of such plots are shown in Figure 6. Note: The two plots are for different trials - and there is no correspondence of solution numbers between the two plots. The figure on the left plots connectivity cost vs. accessibility penalty for a set of solutions. The first example solution would lie closer to the bottom right of the Pareto surface (lower connectivity cost and higher accessibility penalty) compared to the second example which would like on the top left. Neither of the solutions dominates the other and both lie along the same Pareto frontier.

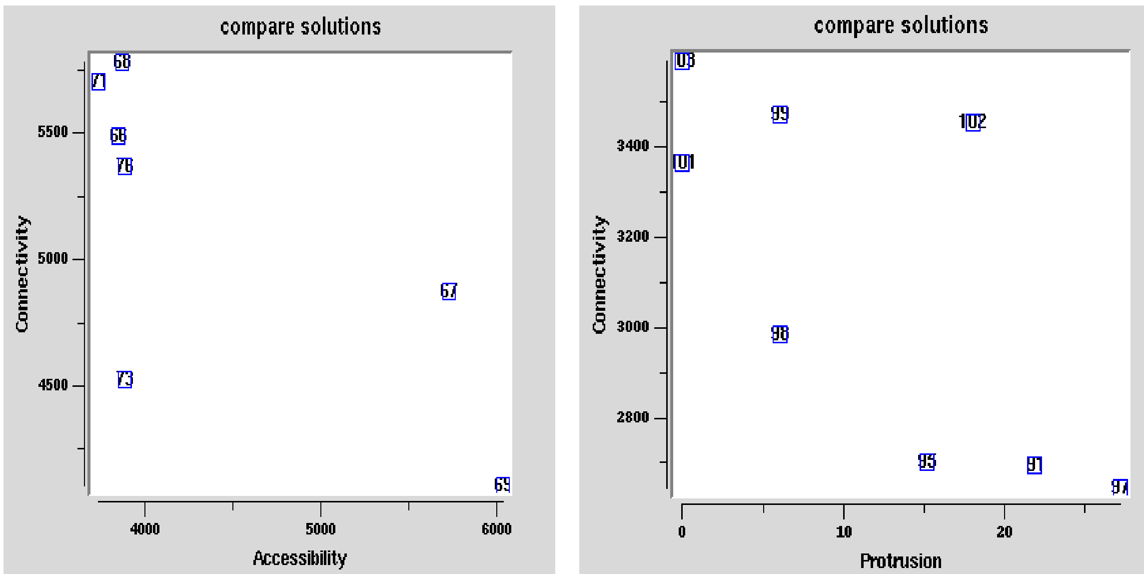


Figure 6. Pareto Frontiers - Comparing solution quality in the A-Team Spatial Layout Tool

4. SUMMARY AND FUTURE WORK

We have developed an A-Team based tool for Spatial Layout. The tool is modular, with separate agents for each constraint. To adapt the solution approach to a specific problem instance, the tool can be easily reconfigured (even during execution) by removing or adding agents to the team. We have demonstrated that a set of independent agents working in parallel on small subsets of the constraints can be organized to solve problems with complicated geometry.

We developed the tool using a set of random packing problems and tested it with a spatial layout problem representing an electromechanical assembly with 10 components, a highly non-convex housing, and 5 constraints. We are extending the test set to include more practical examples such as the layout of the components under the hood of a car.

The tool is also open to the use of different heuristics. It allows the exact evaluation based heuristics to be used alongside approximation based ones. We are extending the set of agents to include the use of evaluation based heuristics.

The A-Team paradigm facilitates interaction and cooperation between humans and computational agents to solve difficult problems. This ability is exceptionally useful for Spatial Layout problems since human beings are very good at identifying and distinguishing promising from unpromising layouts. In a significant number of cases, humans are much better than the best algorithms available. Computers, on the other hand, are much better at numeric processing. We are extending the tool to allow for synergistic cooperation, harnessing the pattern recognition ability of the human with the numerical abilities of the machine. Other extensions to the tool include the use of alternate representations for objects and solutions, allowing rotational freedom, upgrading from ACIS 2.1 to ACIS 3.0 class libraries, and addition of diversity (e.g. by adding simulated annealing based agents) to the team.

5. ACKNOWLEDGEMENT

This research was funded in part by DARPA under contract ONR #N0014-96-1-0854, and by the Institute for Complex Engineered Systems at Carnegie Mellon University.

REFERENCES

[Aziz91] N. M. Aziz, "A computer-aided box stacking model for truck transport and pallets," *Computers in Industry*, v17, 1991, pp. 1-8.

[ALLP97] E. H. L. Aarts, P. J. M. van Laarhoven, C. L. Liu, Peichen Pan, "VLSI layout synthesis", in [AL97] Ch 12, 1997.

[Baker74] K. R. Baker, *Introduction to Sequencing and Scheduling*, Wiley, NY, 1974.

[Chen92] C.L. Chen, "Bayesian Nets and A-Teams for Power System Fault Diagnosis," Ph. D. dissertation, Electrical and Computer Engineering Department, Carnegie Mellon University, Pittsburgh, PA, 1992.

[CTS93] S. Y. Chen, S. N. Talukdar, N. M. Sadeh, "Job-Shop-Scheduling by a Team of Asynchronous Agents," *IJCAI-93 Workshop on Knowledge-Based Production, Scheduling and Control*, Chambery, France, 1993.

[Davis91] L. Davis (editor), *Handbook of Genetic Algorithms*, Van Nostrand Reinhold, 1991.

[DD92] K. A. Dowsland, W. B. Dowsland, "Packing Problems," *European Journal of Operational Research*, v.56, 1992, pp 2-14.

[DeSouza93] P. de Souza, "Asynchronous Organizations for Multi-Algorithm Problems," Ph. D. dissertation, Dept. of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA, 1993.

[Dowsland93] K. Dowsland, "Some Experiments with Simulated Annealing Techniques for Packing problems," *European Journal of Operational Research*, v68, 1993, pp. 389-399.

[GHSTM] S. R. Gorti, S Humair, R. D. Sriram, S. Talukdar, S. Murthy, "Solving Constraint Satisfaction Problems Using A-Teams," to appear in *AI-EDAM*.

[Glover89] F. Glover, "Tabu Search-Parts I and II," *ORSA Journal of Computing*, Vol. 1. No. 3, Summer 1989 and Vol. 2, No. 1, Winter 1990.

[GMM90] H. Gehring, K. Menschner, M. Meyer, "A Computer Based Heuristic for Packing Pooled Shipment Containers," *European Journal of Operational Research*, v44, 1990, pp. 277-288.

[Hinzman95] B. Hinzman, "The Personal Factory," *Rapid Prototyping Journal: Internet conference. Dec. 1995. ISSN 1355 2546*. http://www.mcb.co.uk/services/conferen/dec95/rapidpd/hinzmann/backgnd7.htm

[HT90] R. W. Haessler, F. B. Talbot "Load Planning for shipments of low density products,", *European Journal of Operational Research*, v. 44, 1990, pp. 289-299.

[IBKRW97] I. T. Ikonen, W. E. Biles, A. Kumar, R. K. Ragade, J. C. Wissel, "A Genetic Algorithm for Packing Three-Dimensional Non-Convex Objects Having Cavities and Holes," *Proceedings of the 7th International Conference on Genetic Algorithms*, Michigan State University, East Lansing, MI, 19-23 July, 1997. http://www.spd.louisville.edu/~itikon01

[KGC83] S. Kirkpatrick, C.D. Gelatt, and M.P. Cecchi, "Optimization by Simulated Annealing," *Science*, Vol. 220, Number 4598, May, 1983.

[KMK91] T. Kawakami, M. Minagawa, Y Kakazu, "Auto Tuning of 3-D Packing Rules Using Genetic Algorithms," *IEEE/RSJ International Workshop in Intelligent Robots and Systems IROS '91*, Osaka, Japan, pp. 1319-1324.

[Lengauer90] T. Lengauer, *Combinatorial Algorithms for Integrated Circuit Layout*, John Wiley & Sons, Chichester (England), 1990.

[LGTH97] J. A. Lukin, A. P. Gove, S. N. Talukdar and C. Ho, "Automated Probabilistic Method for Assigning Backbone Resonances of (^{13}C, ^{15}N)-Labelled Proteins," *Journal of Biomolecular NMR*, 9, 1997.

[LMHM95] H. Lee, S. Murthy, W. Haider, D. Morse, "Primary Production Scheduling at Steel making Industries," IBM report, 1995

[Murthy92] S. Murthy, "Synergy in Cooperating Agents: Designing Manipulators from Task Specifications," Ph.D. dissertation, Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA, 1992.

[Quadrel91] R.W. Quadrel, "Asynchronous Design Environments: Architecture and Behavior," Ph. D. dissertation, Department of Architecture, Carnegie Mellon University, Pittsburgh, PA, 1991.

[RWMTSAFAYHJ96] J. Rachlin, F. Wu, S. Murthy, S. Talukdar, M. Sturzenbecker, R. Akkiraju, R. Fuhrer, A. Aggarwal, J. Yeh, R. Henry and R. Jayaraman, "Forest View: A System For Integrated Scheduling In Complex Manufacturing Domains," IBM report, 1996.

[SC93] S. Szykman, J. Cagan, "Automated Generation of Optimally Directed Three Dimensional Component Layouts," *Advances In Design Automation*, DE-Vol. 65-1, 1993, pp. 527-537.

[SC95] S. Szykman, J. Cagan, "A Simulated Annealing-Based Approach to Three-Dimensional Component Packing," *Transactions of the ASME, Journal of Mechanical Design*, v. 117, June 1995, pp. 308-314.

[Szykman95] "Optimal Product Layout Using Simulated Annealing," Ph.D. Dissertation, Mechanical Engineering Department, CMU, Pittsburgh, 1995.

[Talukdar98] "Autonomous Cyber Agents: Rules for Collaboration", Proceedings of the thirty-first Hawaii International Conference on System Sciences (HICSS-31), Hawaii, January 1998.

[TBGD97] S. Talukdar, L. Baerentzen, A. Gove, P. de Souza, "Asynchronous Teams: Cooperation Schemes for Autonomous Agents", to appear in *Journal of Heuristics*, 1997.

[TPG83] S. N. Talukdar, S. S. Pyo and T. Giras, "Asynchronous Procedures for Parallel Processing," *IEEE Trans. on PAS*, Vol. PAS-102, NO 11, Nov. 1983.

[TR93] S. N. Talukdar, V.C. Ramesh, "A Parallel Global Optimization Algorithm and its Application to the CCOPF Problem," *Proceedings of the Power Industry Computer Applications Conference*, Phoenix, May, 1993.

[TSC97] S. N. Talukdar, S. Sachdev, E. Camponagara, "A Collaboration Strategy for Autonomous, Highly Specialized Agents," *Proceedings of the SPIE*, Symposium on Intelligent Systems & Advanced Manufacturing, October 1997.

[Tsen95] C. K. Tsen, "Solving Train Scheduling Problems Using A-Teams," Ph.D. dissertation, Electrical and Computer Engineering Department, CMU, Pittsburgh, 1995.

[WF94] J. R. Wodziak, G. M. Fadel "Packing and Optimizing the Center of Gravity Location using a Genetic Algorithm," http://www.vr.clemson.edu/dmg/papers/Opt-papers.html, Design Methodology Group, Clemson University, Clemson SC.