To appear in the Proceedings of the
1997 International Conference
on Robotics and Automation

Agile Assembly Architecture: An Agent Based Approach to Modular Precision Assembly Systems

Alfred A. Rizzi, Jay Gowdy, and Ralph L. Hollis

The Robotics Institute
Carnegie Mellon University

Abstract

This paper serves to present our initial thoughts on the design of an architecture for highly flexible, modular, and distributed precision assembly systems. Through the use of a unified design, simulation, programming, and monitoring environment coupled with self-representing cooperative agents this architecture will significantly simplify the process of factory design and deployment. Our demonstration of this architecture takes the form of a "table top" sized assembly system targeted at the partial assembly of high-density magnetic storage devices.

1 Introduction

As part of a multi-million dollar four-year project funded under the NSF Multi Disciplinary Challenges component of the High Performance Computing and Communication program, this paper documents our initial thoughts on the development of a miniaturized, modular, high-precision assembly system or *minifactory*. We believe this type of system can only be developed through careful integration of hardware and software tools in a manner heretofore unseen in the robotics and automation community. Specifically we wish to construct a distributed system of tightly integrated mechanical/computational agents endowed not only with information about their own capabilities but also with the ability to appreciate their role in the factory as a whole and negotiate with their peers in order to participate in flexible factory level cooperation. It is the formalization of these basic ideas that constitutes our notion of an *Agile Assembly Architecture* (AAA).

AAA will use factory-wide standard procedures, protocols, and well structured agent autonomy to simplify the process of designing and programming complex high-precision assembly systems. The unified design and programming tools of AAA will allow a user to select agents over the Internet and program them in a simulated factory environment. More importantly, AAA will take advantage of agents' self-knowledge and ability to explore their environments to make the transition between simulation and reality as painless and seamless as possible.

To facilitate the design and operation of distributed systems of this type, every agent in a minifactory will contain its own computer capable of both representing itself to its peers and providing detailed models (both geometric [1] and behavioral) for use in its simulation. To support a suitably extensible simulation environment we expect to make use of distributed models and processing by relying on agents (and their associated computers) to represent their own behavior during simulation. This mitigates the need to provide necessarily restricted models for agent behavior and allows for simplified integration of new types of agents into AAA. One positive and intentional side-effect of this approach is that it will be possible to remotely access (via the Internet) agents which a developer is considering for inclusion in a factory and "use" them in a simulation of the system.

Our hope is that AAA and minifactory can provide viable high-precision assembly alternatives for industries that would benefit from drastically reduced factory design and deployment time. The aim of the current project is to demonstrate the application of this assembly technology to the partial assembly of high-density magnetic data storage devices. More generally, minifactory is suitable for use in the production of short lead time products consisting of moderate sized high-precision parts and assemblies.

Section 2 of this paper offers a brief overview of the capabilities and advantages central to the minifactory. Section 3 presents an overview of the issues involved in the development of the design, simulation, and monitoring tools we foresee being necessary to support the rapid design, deployment, and programming of minifactory systems. Section 4 addresses the architectural requirements placed on the individual agents to allow their smooth integration into a minifactory system. Finally, Section 5 offers a brief overview of the robotic sensing and motion technologies under development in the Microdynamic Systems Laboratory[1] for use in the current project.

[1] See `hhtp://www.cs.cmu.edu/~msl`

2 What is a Minifactory?

We have deliberately chosen the scope of capabilities we wish a minifactory to perform to afford both analytic tractability and design practicality. Toward this end we have limited the class of tasks we will perform to assembly and processing operations requiring four or fewer degrees of freedom. Specifically we want to construct systems capable of:

- Four-degree-of-freedom vertical insertion.
- Easily integrating overhead processing (*e.g.* laser processing or material/glue deposition).
- Micron-level part placement accuracy.
- Factory design and programming in less than a week.

To provide this functionality, a minifactory consists of a potentially large collection of mechanically, computationally, and algorithmically distributed agents. Each agent in this collection is responsible for providing a minimum level of cooperation and communication in order to participate in the most basic minifactory operations, thereby simplifying the development and deployment of the system.

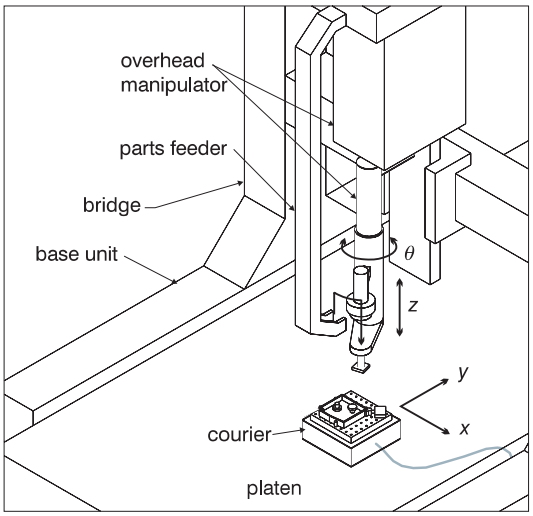

Figure 1: Basic components of a minifactory.

The most obvious departure from traditional automation systems and one of the most obvious embodiments of our philosophy of factory level integration can be seen in our choice to integrate product transfer and local manipulation. As such, we have eschewed the traditional use of SCARA manipulators coupled with part conveyor systems and local fixtures. Alternatively, as depicted in Figure 1, we have chosen to make use of two-degree-of-freedom (DOF) manipulators (MANIPULATORS) and two-DOF planar manipulators (COURIERS) moving over a high-precision platen surface. The COURIERS are thus responsible both for product transport within the factory and for transiently forming cooperative four-DOF manipulators when they present sub-assemblies to a stationary MANIPULATOR.

This approach to providing a four-DOF capability has a number of advantages [6, 8]:

- **Precision:** Joint flexibility, link flexibility, sensor precision, etc. all limit the accuracy of serial linkage based systems.
- **Speed:** Lower masses of the individual manipulators affords greater acceleration. Additionally a MANIPULATOR can begin to prepare for the arrival of a new COURIER while the COURIER it had been interacting with is performing the next operation on its product.
- **Flexibility:** Mechanical and electrical modularity allows easy integration of semi-custom processing elements (*e.g.* screwdrivers, orbital head formers, glue dispensers, laser processors, etc.).

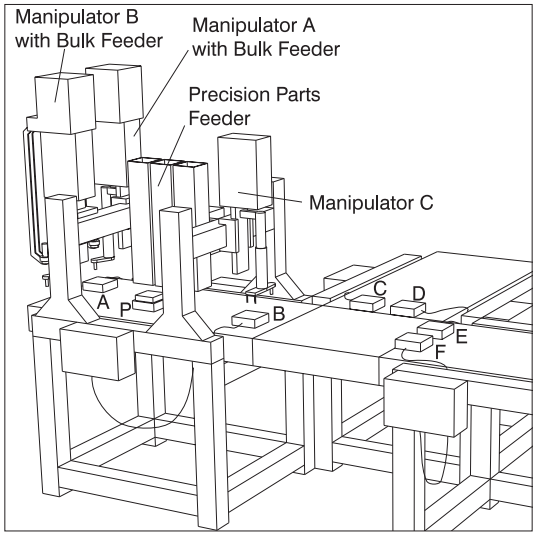

Figure 2: View of a "typical" section of a minifactory assembly system, including a "tee" junction.

Perhaps the best way to appreciate the implications of this approach to the design of factory level assembly systems is to consider a somewhat contrived but illustrative example. Figure 2 depicts a view of a small section of a fictitious minifactory. The system pictured includes six COURIERS and three MANIPULATORS with two bulk-random parts feeders and one precision feeder. COURIERS begin on the left of this system,

present their sub-assemblies to the first two manipulators where two new components are added; the resultant sub-assemblies then travel to the right where the final manipulator is responsible for both placing a precision component and transferring the final assembly to one of four COURIERS.

3 User-Level Design, Programming, and Monitoring Tools

A minifactory program is not a centralized list of commands describing the global operation of the assembly system. Rather, each agent, whether it be a COURIER, MANIPULATOR, or other custom robot, is an independent entity executing its own program. The overall minifactory behavior results from each agent interacting both with its environment and peers. The central challenge for the minifactory simulation and programming environment is to facilitate the development of well debugged distributed programs, while simultaneously easing the difficult transition from the simulated world of bytes and pixels to the real world of actuators and sensors.

In the absence of a centralized controller, a minifactory will have a centralized user *interface tool* capable of supporting the design, simulation, and run-time monitoring and control of a minifactory.

3.1 Design and Programming

The goal of the minifactory programming environment is to simplify the difficult problem of integrating the components of a factory and generating the detailed, interacting programs for every agent in the system. A minifactory system should be designed and programmed by an expert in the assembly problem at hand without requiring expertise in minifactory programming.

Following current trends in the field of machine programming, *e.g.* [5], the design and programming of a minifactory will be primarily graphical, with as little text interaction as possible. Users should be able to point, click, and drag their way through an intuitive visual interface to create a distributed minifactory program. Within this framework we foresee the use of constraints, such as local frames of reference, and abstractions, such as coordinating the gross motion of COURIERS through the use of distributed resource management rather than considering it as a global allocation problem. Such a use of constraints guides the user to construct robust systems while abstractions hide details that the user cannot afford to be concerned with if correct programs are to be rapidly built.

The most significant constraint in a minifactory program is that all geometric references will be made with respect to physical components of the factory, such as MANIPULATORS or feeders. Thus, programs become naturally robust to minor variations in the geometric configuration of an actual factory, presuming that the topological relationships between factory components are maintained.

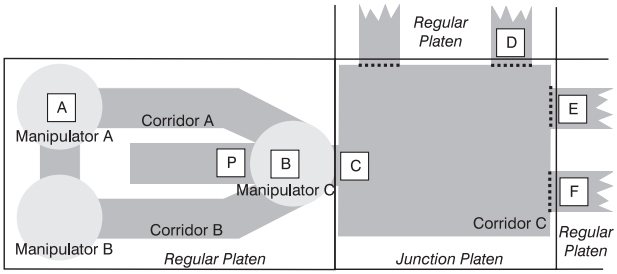

Figure 3: Schematic example of reservation areas for the factory of Figure 2.

We abstract the management of the point-to-point motion of agents through the use of a geometry reservation system. Agents are free to move in previously reserved areas, which are $2\frac{1}{2}$D volumes in which an agent and its cargo are guaranteed to be safe from collision with other components of the minifactory. As with all geometric references in a minifactory program, the geometry of a reservation area will be be anchored to various elements of the factory. For example, as shown in Figure 3, Corridor A forms a reservation area with ends anchored to Manipulator A and Manipulator C. Thus, a user can specify the gross movement of a COURIER by selecting (or creating) reservation areas in sequence, interspersed with "rendezvous" areas (*e.g.* the areas under the MANIPULATORS in Figure 3), where the COURIER will interact with other agents.

This approach to geometry management and gross motion planning provides several advantages to the user:

- **Abstraction:** The user does not have to specify that a specific COURIER must explicitly contact some other COURIER and/or MANIPULATOR for permission to move into a manipulator's workspace; all of these negotiations are hidden through the use of a reservation area.
- **Modularity:** Rather than a MANIPULATOR knowing it has to interact with a particular COURIER, it just has to know that it interacts with whatever COURIER has reserved its workspace.
- **Robustness:** Since the reservation areas are referenced to physical components of the factory, if these components move slightly, the various agent programs will continue to function properly.

3.2 Simulation

Minifactory simulations allow a user to explore the application of minifactory technology to a particular assembly problem, and to do much of the development and debugging of the factory programs off line in a virtual environment. A key to a minifactory's rapid and successful deployment is the rapid and successful transition of a factory program from this virtual environment to the actual machines. The two facets of our architecture that make this transition possible are *fidelity* of the simulation and *robustness* of the underlying agents. Fidelity demands that the simulated factory will behave identically to an actual factory configured in exactly the same way. Robustness of the agents acknowledges the inability to configure an actual factory in exactly the same manner as it was simulated, but that we can detect and account for the differences.

We achieve fidelity through the agents' self knowledge and self representation. Each minifactory agent provides a representation of its own geometry, behavior, and integration constraints. Thus, the simulator will not use a catalog to look up the characteristics of a typical MANIPULATOR, but rather will query an actual MANIPULATOR via an Internet connection for its own self-representation. This reliable representation of an agent's characteristics will eliminate many inaccuracies that would otherwise occur. Robustness is provided by the ability of the agents to self-calibrate and explore their environment (as described in Section 4.2). Since geometric references are anchored to features in the environment, the effects of any discovered differences between the actual factory configuration and the simulated factory simulation are eliminated.

Additionally we foresee simplifying the transition from simulation to reality by allowing mixed operation of the simulation system in conjunction with running hardware. In full simulation mode, most of the agent models and programs will be internal to the simulation environment itself, each of them having been constructed from the description provided by the agents themselves. In practice there is no reason beyond efficiency why the implementation of these agent models could not be performed by the remote agents themselves rather than internal to the simulator. Thus, simply by mixing internal agent models and external agents, a real agent could be put through its paces in isolation, with all its actuators working and sensors gathering data, but within the context of a greater simulated system.

3.3 Real-time interaction

The interface tool will not just support the design and simulation of a factory — it will also monitor an operating factory. Rather than serve as a central "brain" for the system it will act as a clearing house for status, errors, and product information.

The interface tool's primary role in a running minifactory is to present views, on several levels, of what is currently happening in the factory as reported by the agents. The major view will be an annotated 3D representation of the factory as a whole, showing the product flow through the system. The user will also be able to select any agent in the system and bring up its "control panel." The content of these control panels will range from concrete physical properties such as joint angles and accelerations to abstract behavioral properties such as programs and errors in achieving goals, but the specific content and how these views are updated is determined by the agents themselves rather than the interface tool. The self description of the control panel provides a level of flexibility and extensibility that would be impossible if the interface tool just had standard control panels for standard components.

Beyond passively presenting factory and agent operations to the user, the interface tool will actively monitor the state of the whole system to look for problems and conflicts. These problems could be physical, such as collisions between COURIERS, but more commonly they will be "logical," such as overlapping reservation areas. In addition, the interface tool will monitor and store product information such as part counts, defect rates, flow rate, etc. This information can be stored for later analysis both to improve the assembly plan in the future and as a record of how the assembly process went.

4 Run-Time Coordination and Communication

Any minifactory agent, be it a COURIER, a MANIPULATOR, or some custom designed module, must provide a minimal level of capability in order to work in the minifactory. Currently we foresee there being four general classes of capability every agent must reliably provide: basic trustworthiness, self initialization, resource negotiation, and inter-agent coordination.

4.1 Basic Trustworthiness

For an agent to be a successful member of a factory community its peers must be able to trust it to reliably represent itself. Practically, we see this manifesting itself in the form of three fundamental capabilities.

- All agents must advertise their basic capabilities and the protocols they understand to their peers.
- Every agent must be capable of reporting its current status and location.
- Each agent must implement reliable and safe failure detection and recovery schemes.

The first two of these requirements are essential to support graceful coordination between minifactory agents, their peers, and factory monitoring tools. The

need to advertise capabilities further implies the existence of a predefined extensible protocol suitable for the exchange of such information between agents. The next two capabilities may well be the most important, and quite possibly the most difficult to precisely define and implement. The assertions demand that agents be capable of constantly monitoring the state of the factory available to them. Furthermore, when an agent detects conditions outside the norm it must be capable of independently correcting the aberration, negotiating with its peers to recover from the fault, or broadcasting its inability to proceed thus bringing the factory to an orderly stop. Although it is potentially difficult to guarantee this level of capability in an arbitrary agent we feel that through judicious use of a combination of traditional AI reasoning [7] and reactive behaviors [4] that it can be achieved in the highly-constrained domain of minifactory.

4.2 Factory Calibration/Initialization

Integral to the rapid deployment of an assembly system is the need for precise and automatic calibration and initialization whenever a factory is "turned on." There are three interrelated tasks that must be collectively undertaken by the minifactory agents to successfully initialize a factory system. This process will begin with agents identifying their peers through the use of messages broadcast to the factory at large. Following this, COURIERS must explore their environs to discover both the exact geometry of the platen surfaces, as well as the positions of any stationary agents within their range. Finally, through a careful exchange of this information between agents, a complete map of the minifactory can be constructed both in the agents and the monitoring interface tool. At this point, the interface tool can distribute programs developed for simulated agents in a simulated system to the appropriate real agents in the minifactory and those programs can be verified and registered against the newly discovered geometry of the minifactory.

It would be impractical, if not impossible, to assemble a real minifactory that corresponds to within microns of an equivalent simulated minifactory. Without self calibration and exploration, transferring programs from the simulation environment to the real minifactory would involve the painstaking process of rewriting and calibrating all of the agent programs. With self-calibration and local frames of reference, agents can accommodate minor variations in factory construction and detect major variations which might invalidate their programs.

4.3 Resource Negotiation

As introduced in Section 3, we chose to make individual agents both capable of and responsible for negotiating for the use of shared resources. This has significant implications for the gross motion of agents, as the primary resource in contention will be space on the platen surfaces. To both simplify factory level programming and ensure safe operation, we are proposing the use of a distributed reservation-based scheme to resolve these types of conflict. This approach requires that we make the assumption that an agent will only go where it says it will go, and that there are no "outside" influences which fail to reserve the resources they consume. These assumptions — which are reasonable in the highly structured, very stable, and well known minifactory environment — allow us to dispense with the inter-agent perception systems that would be necessary to implement completely "reactive" motion, in which agents would be required to observe other agents' positions and intentions (either with sensors or by querying) prior to taking action. The low cost and predictable behavior obtained through the use of a reservation system far outweighs the risk of our assumptions being violated and the minor efficiency losses which will inevitably be incurred.

We also foresee using a similar distributed reservation system to arbitrate the consumption of more abstract resources such as vibration, noise, thermal, or optical emissions. The use of these reservation schemes depends on the trustworthiness of agents, *i.e.* that they can both know and advertise what resources they require and are consuming.

4.4 Agent Coordination

Critical to the successful application of a distributed machine of this class is the capability for agents to reliably interact and coordinate their activities. We envision these local interactions taking place through standardized protocols between small groups of agents (typically two). The most fundamental form of this cooperation will happen whenever a COURIER and MANIPULATOR transiently form a four-degree-of-freedom system to perform a part placement task. The most basic mode for such cooperation will take the form of a virtual linkage between two machines where one agent is effectively slaved to the state of the other, allowing for simple coordinated movement. Other modes of cooperation will include coordinated behavior changes and cooperative sensor-based alignment. Behavior changes would be used to encode the sequence of operations necessary for a high-precision force-controlled insertion task (*e.g.* MANIPULATOR exerts low vertical force while COURIER "finds" the hole, followed by the COURIER becoming compliant while the MANIPULATOR exerts higher forces to perform the insertion).

5 Robotic Sensing and Motion Technologies

In parallel with the development of architectural approaches to multi-agent coordination and factory de-

sign presented in Sections 3 and 4 we are also constructing machines of the form depicted in Figures 1 and 2.

The first and possibly most critical technology under development is the precision control and sensing for COURIERS [9]. Toward this end, we are pursuing two potential local sensing technologies, one magnetically based, the other optical. In principle, either can deliver the needed micron level resolution. A prototype of the magnetic sensor is currently under evaluation, and we expect to evaluate a prototype of the optical sensor in the near future [3].

Unfortunately COURIERS require tethers for the supply of both power and air. This both complicates motion planning for the COURIERS as well as forcing the inclusion of additional mechanisms and fixtures to support the transfer of partial assemblies between COURIERS. Implicitly this results in a limit on the total number of couriers which can be active in a particular region of a minifactory thus imposing an upper bound on the overall efficiency and throughput of a system.

Concurrent to the development of COURIERS, work is being done to design both the MANIPULATORS and parts feeding subsystems. While the design and construction of a MANIPULATOR may not seem particularly complex, there are a number of issues which must be addressed to achieve the desired positioning resolution and accuracy. Not surprisingly, the design of general purpose part feeding systems is quite complex and has been the topic of significant independent research [10, 2]. Our project is beginning to explore potential solutions that fit our application needs.

Finally, it is critical to provide suitably scalable computational and network resources throughout the minifactory environment for both user commands/monitoring, and inter-agent coordination. We have chosen to provide these capabilities through the use of two network ports per agent: one for factory wide non-real-time communication, and the second for local real-time coordination and communication.

6 Conclusion

This paper represents a first effort to introduce our vision of the future of high precision assembly, we see this distributed approach to factory design, modeling and control affording a number advantages:

- **Modularity:** Segments of a minifactory can be modified or expanded with minimal impact on neighboring parts of the factory.
- **Robustness:** Mechanical and computational distribution allows for natural local error detection and recovery.
- **Scalability:** Congruency between the computational and mechanical systems removes the traditional bottlenecks associated with a centralized model of factory control. In fact the information flow and programs are as distributed as the actual product flow leading to a natural symmetry between their management.
- **Ease-of-use:** Unified design and development tools in conjunction with capable agents will drastically simplify the process of designing, programming, and debugging an assembly system.

Acknowledgments

This work is supported in part by NSF grants DMI-9523156 and CDA-9503992, and by the CMU Engineering Design Research Center.

References

[1] ISO 10303. STEP: Standard for the Exchange of Product model data.

[2] Geoffrey Boothroyd, Corrado R. Poli, and Laurence E. Murch. Handbook of feeding and orienting techniques for small parts. Department of Mechanical Engineering, University of Massachusetts, Amherst MA, 1976.

[3] A. E. Brennemann and R. L. Hollis. Magnetic and optical-fluorescence position sensing for planar linear motors. In *Proc. Int'l Conf. on Intelligent Robots and Systems*, volume 3, pages 101–107, Pittsburgh, August 1995.

[4] R. R. Burridge, A. A. Rizzi, and D. E. Koditschek. Sequential composition of dynamically dexterous robot behaviors. submitted to the International Journal of Robotics Reaserch, August 1996.

[5] M. W. Gertz, D. B. Stewart, B. J. Nelson, and P. K. Khosla. Using hypermedia and reconfigurable software assembly to support virtual laboratories and factories. In *Proc. 5th Int'l Symp. on Robotics and Manufacturing*, Maui, Hawaii, August 15-17 1994.

[6] R. L. Hollis and A. Quaid. An architecture for agile assembly. In *Proc. Am. Soc. of Precision Engineering, 10th Annual Mtg.*, Austin, TX, October 15-19 1995.

[7] D. J. Musliner, E. H. Durfee, and K. G. Shin. World modeling for the dynamic construction of real-time control plans. *Artificial Intelligence*, 74(1):83–127, March 1995.

[8] A. Quaid and R. L. Hollis. Cooperative 2-DOF robots for precision assembly. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, Minneapolis, May 1996.

[9] A. E. Quaid, Y. Xu, and R. L. Hollis. Force characterization and commutation of planar linear motors. In *Proc. IEEE Int'l Conf. on Robotics and Automation*, April 1997.

[10] Jeff Wiegley, Ken Goldberg, Mike Peshkin, and Mike Brokowski. A complete algorithm for designing passive fences to orient parts. In *Proceedings IEEE International Conference on Robotics and Automation*, pages 1133–1139, April 1996.