# Cooperative and Coordinated Assembly for Heterogeneous Robots via Distributed Mobile Agents

Juan Rojas* Electrical and Computer Engineering Department
Center for Intelligent Systems
Vanderbilt University
Nashville, TN 37235, USA
rojas70@gmail.com

Richard A. Peters II** Electrical and Computer Engineering Department
Center for Intelligent Systems
Vanderbilt University
Nashville, TN 37235, USA
rap2@vuse.vanderbilt.edu

***Abstract*— Robotics technology is quickly evolving and demanding robots to perform more actions and with greater complexity. Some tasks must be executed through teams of homogeneous or heterogeneous teams of robots. Complex robotic systems are making used of distributed multi-agent architectures to facilitate the development, integration, and deployment of such systems. Similarly, modular control is playing an important role in rendering more flexible and adaptive controllers for complex systems. This paper presents a team of heterogeneous robots performing an autonomous and collaborative assembly task that is grounded on a distributed multi-agent architecture and modular control basis approach. We conducted experimental results to assess the efficacy of the system and concluded that multi-agent multi-robotic collaborative systems with modular control approaches is a viable approach to generate complex robotic behavior.**

## I. INTRODUCTION

Robotics technology is quickly evolving and demanding robots to perform more actions and with greater complexity. Such robot systems need to perform those tasks with increased flexibility and autonomy [1]. Robotics is too moving towards multi-robotic systems in which robots of different morphologies collaborate or coordinate with one another [2].

In order to achieve the implementation of complex robotic systems, researchers have experienced that distributed multi-agent architectures or "mobile agents" has facilitated the development, integration, and execution of such systems [3]. Multiple frameworks have been built over the last decade in an effort to facilitate the development of robotic systems. Mobile agents usually seek to abstract complex behavior through a hierarchical taxonomy of low-level to high-level primitives [4], supported by strong encapsulation of code to allow fore modularity, scalability, and reusability. Distributed architectures allow to implement hardware and software mechanisms across different computers, which is an increasingly important attribute as more robotic systems contain more than one computer on-board [5].

Similarly, complex robotic systems are making use of adaptive and flexible controllers to attenuate the limitations posed by monolithic controllers and unstructured environments [6]. In [7], an adaptive control strategy used observers and parameter update laws to estimate the stiffness and geometry of objects in the environment for a 2 DoF manipulator in simulation. In [8], a self-tuning proportionalintegralderivative (PID) controller scheduled tasks adaptively for a real-time task scheduler.

This paper seeks to assess the viability of this implementation under a multi-robot, collaborative assembly task, with flexible and adaptive controllers. To this end, the authors deployed an agent-based distributed robotic system to perform a collaborative assembly task through the use of an also modular robust and adaptive control approach known as the control basis approach. The modularity and the flexibility of the control approach works in concert with modular agents to bootstrap robust but flexible controllers that generate a reactive and autonomous system. The authors chose to task two heterogenous robots: an anthropomorphic dual-armed, six DoF, and pneumatically actuated robot ISAC [9] and a rigid, point-to-point, six DoF industrial manipulator HP3JC robot to perform a collaborative assembly task. To test the flexibility of the system, the task was executed by having robots switch roles. For the first experiment, the HP3JC robot would serve as a pusher while ISAC would serve as a holder, and for the second experiment those roles would be reversed.

## II. THE INTELLIGENT MACHINE ARCHITECTURE

The intelligent machine architecture was developed at the Center for Intelligent Systems at Vanderbilt University with the governing principles of being a decentralized agent architecture that would facilitate the development, integration, and execution of complex robotic systems [4], [10]. To this end, the architecture was designed to be decentralized, multilingual, scalable, and reusable. The architecture offers a well-defined agent model, runtime environment, data connection, and user development tools.

The agent model can implement atomic components that encapsulate hardware or sensor resources, skills or behaviors, an environment, and finite state machines (FSMs). These atomic components can be combined in a hierarchical tree to form compound agents. An example could be a visual agent composed on multiple atomic components that encapsulated frame grabbing, image buffers, color segmentation, and tracking behaviors amongst others. The agent model follows a taxonomy by which components are divided into four categories: (a) Mechanisms - they represent sensor or hardware resources, skills or behaviors; (b) Representations - they represents visual, auditive, or numeric data; (c) Engines

- they represent event-driven state machines; and (d) Agents - a hierarchical component able to be comprised of multi-type components.

The IMA Architecture uses Microsoft's DCOM as its communication protocol. The agent model interfaces to the architecture's runtime environment though simple IDE interfaces. The runtime environment is composed of distributed layer, a control layer, and an application layer. The distributed layer is responsible for handling the messaging that takes place across components within or across computers. The control layer maintains an organized record of all components in the system, and the application layer provides the End-User interfaces and development tools. IMA counts with an intuitive Agent Construction tool - Distributed Agent Designer (DAD), a visual debugging tool - Manager Book, and a low-level command-line debugging tool - Command Console. DAD is an GUI where one can create new components or reuse existing ones. Component interface parameters can be modified on- or off-line for testing and system verification procedures. Lastly, FSMs can be manually or autonomously triggered in DAD.

IMA was used to create hundreds of components, tens of agents, across six computers and 2 robots. The architecture encapsulated low-level behavior by controlling sensors as varied as frame grabbers, force-torque sensors, encoders, pan-tilt actuation, servo motor actuation, pneumatic actuation, and open-and-closing grasps. Middle-level abstractions included control basis controllers, visual tracking, path planning, homing routines, and object recognition. All of these middle-level abstractions were initialized, triggered, and stopped through the use of event-driven finite state machines. At the highest level there were 'brain' like agents that oversaw the smooth execution of the assembly strategy according to the roles enacted by each robot (these had to be chosen *a priori* as learning was not part of this demonstration).

## III. THE CONTROL BASIS APPROACH

A control basis decomposes a complex control system into a set of modular control elements that when connected appropriately synthesize a variety of behaviors. A control basis consists of any number of closed loop controllers (that represent primitive actions) derived from a set of control laws. As asserted by Huber [6], the control laws are designed to yield asymptotically stable and predictable behavior for different robots. That is, each carefully selected control law is designed to be robust under a wide range of conditions and largely independent of robot kinematics. Each control law discretizes the continuous space into discrete basins of attraction. The control law can compensate for a limited range of perturbations and uncertainties while still converging to the attractor. Similar approaches are found in the literature [11], [12], but the control basis framework is different in that it factors controllers into *objectives* and can combine any number of controllers through the use of nullspace compositions in any order to achieve a wide range of behaviors. A careful selection of a small set of control laws is realized as controller objectives implemented through the use of selected sensor and actuator resources to produce flexible structural solutions. In effect, the approach allows control elements to be re-used and generalized to different solutions depending on the context [13]. A solution that works in concert with mobile agent paradigms.

### A. *Mathematical Derivation*

We begin by describing a primitive closed-loop controller and subsequently detailing a methodology to combine and optimize the results of two controllers (primitive or compound). For our system, two robot manipulators with 6 DoF are used, where $q \in \Re^6$ is a vector of joint angles for both manipulators, and $x \in \Re^6$ is the vector of position and orientation for a robot's end-effector.

Primitive controllers $\phi_i$, where $i = 1 \sim n$, are elements in a basis of controllers, $\Phi$, such that $\phi_i \in \Phi$. A primitive controller optimizes a partitioned portion of a designated control space and can be understood as the minimization of a discrete basin of attraction. The basins of attraction are formulated through artificial potential functions defined over a typed domain (such as cartesian positions), which are defined as the square of the error:

$$\phi_i(\rho) = \rho^T \rho \tag{1}$$

where the error, $\rho$, is the difference between the reference input and the plant input, $\rho = q_{ref} - q_{des}$, at every time step.

Each controller reaches its objective by performing greedy descent, $\nabla\phi_i$, on the artificial potential function, while engaging sensor and motor resources. The minimization of the surface potential function in a specified domain space, $X_i$, is defined as:

$$\nabla_{X_i}\phi_i = \frac{\partial \phi_i}{\partial X_i}. \tag{2}$$

Each primitive is bound to a selected subset of input control resources $\gamma_j \in \Gamma_j$ and output control resources $\gamma_k \in \Gamma_k$ relevant to the task. In order to bind input and output control resources to the controller, corresponding sensor transforms, $s_j$, and an effector transforms, $e_k$, are used. The *sensor transform* maps incoming sensory resources to a specified domain space such that $s_j : \Gamma_j \to X_i$. To ensure that a task is guaranteed to operate within the region of a corresponding basis we require that the output range of a sensor transform matches the artificial potential function domain's data type. Similarly, the *effector transform* maps the control law error's result to an appropriate output space, $Y_k$. The mapping is typically effected using a Jacobian matrix as in Equation 3.

$$e_k(\Gamma_l) = \left( \frac{\partial \phi_{x_{\gamma 1}}}{\partial y_k}, \frac{\partial \phi_{x_{\gamma 2}}}{\partial y_k}, ..., \frac{\partial \phi_{x_{\gamma |S_l|}}}{\partial y_k} \right)^T, \tag{3}$$

where, $\phi_{x_{\gamma 1}}$ represents the controller update for a sensor control resource $\gamma_1$. $y_k$ is a corresponding point in the output space and $\Gamma_l = \gamma_1, \gamma_2, ..., \gamma_{|S_l|}$ is a subset of selected control resources for a given task. The effector too is a function of $\Gamma_l$. In order to match an effector transform with an artificial potential function, the rowspace of $e_k(\Gamma_l)$ must match the potential function's data type.

In conclusion, a closed-loop controller is implemented when the error between the incoming sensor information and the reference position are minimized within the discrete artificial potential basin, $\nabla_{x_i}\phi_i(X_{ref} - s_j(\Gamma_j))$, and having the gradient result mapped onto the output configuration space through an effector transform, $e_k(\Gamma_l)$. Given that the input data is of the same domain type as the artificial potential function, and the effector transform is of the same dimensions as the potential function, the controller's output, $\nabla_{y_k}\phi_i$, is defined as:

$$\nabla_{y_k}\phi_i = e_k(\Gamma_l)^T \nabla_{x_i}\phi_i(\mathbf{x}_{ref} - s_j(\Gamma_j)). \tag{4}$$

For convenience, the above expression is expressed in simplified notation as:

$$\phi_i \mid^{s_j(\Gamma_j)}_{e_k(\Gamma_l)} (\mathbf{x}_{ref}). \tag{5}$$

To concurrently optimize multiple control laws in a system, we use a method originally implemented by Platt [14] based on the Moore-Penrose pseudo inverse to project a secondary control update to the nullspace of the primary objective's equipotential manifold. In this context it is said that within the compound controller $\pi$ the secondary controller $\phi_2$ is *subject-to* the primary controller $\phi_1$, and expressed as:

$$\nabla_y(\phi_2 \triangleleft \phi_1) = \nabla_y\phi_1 + \mathcal{N}(\nabla_y\phi_1^T)\nabla_y\phi_2, \tag{6}$$

where,

$$\mathcal{N}(\nabla_y\phi_1^T) \equiv I - (\nabla_y\phi_1^T)^+(\nabla_y\phi_1^T), \tag{7}$$

and, *I*, is the identity matrix, *y* is an *n*-dimensional space, and $\nabla_y\phi_1^T$ is a (*n*-1) dimensional space orthogonal to the direction of steepest descent [15]. For convenience, Equation (7) is written as:

$$\pi_k : \phi_1 \triangleleft \phi_2 \tag{8}$$

The nullspace operator, $\mathcal{N}(\nabla_y\phi_i^T)$, encompasses a *nullspace composition* technique that optimizes the concurrent execution of two controller. Unlike traditional methods [16], there is no need to specify how control resources will be shared across sub-controllers as long as the same control resources are used.

## B. Control Policy Implementation

Once a number of defined primitives, $\Phi$, sensor transforms, $s$, and effector transforms, $e$, have been determined for a control problem, a policy must be enacted to complete the task out of the combinatoric basis: $\Phi \times 2^s \times 2^e$. By selecting a well defined set of basis controllers, the sequencing of subgoals eases the need for complex control layers or switching criteria. Sequencing of primitive or compound controllers limits the set of tasks that can be addressed and improve predictions across controller sequences [6]. The sequencing, in effect, becomes an instruction set that strings subgoals to achieve an overall task [13].

In this demonstration, the finite state automata consists of two states for the pushing robot and one state for the holding robot. The transition policy amongst these controllers is explained here first and their derivation presented in section IV. For the pushing robot, the first state starts the control composition that advance the male truss towards the female truss. The second state, deploys the insertion of the trusses. For the holding robot, a counter-balancing controller that opposes any motion if presented.

In conclusion, a sequence of concurrently combined controllers encodes an instruction set for complex tasks. The modularization of a control problem in this way prevents monolithic control and reduces the need for complete and accurate system models; thus easing complexity [6]. The enaction of a control policy of this kind using the control basis framework allowed two robots of very different morphologies to perform cooperative assembly tasks flexibly and robustly.

## IV. DEVISING A CONTROL BASIS FOR COOPERATIVE ASSEMBLY TASKS

The assembly task performed by our robots work consists of an insertion where a male truss is inserted into a female fixture. Such an insertion begins by having the male truss positioned at a point in space in front of the female truss through a mobile platform. The male truss may offset from the female truss in all three planes as long as it remains in the workspace of the robot team. The insertion proceeds by having the male truss move in a linear fashion to an optimal insertion point in front of the female truss. To this end, the male truss has an inverted chamfer at the end to simplify the entry, while the female fixture has a cylindrical end of a diameter slightly larger than the truss'. A number of controllers were designed to respond to the push-role consisting of two stages: (a) a guarded approach, and (b) a compliant insertion [17]. For an insertion task to be executed successfully a robot must be able to displace the male truss to an optimum location for insertion. If the final position of a male truss during an approach motion ends at a location outside the interior hole of a female fixture, the assembly cannot succeed.

Similarly, if during the approach, the male truss jams the fixture, a successful insertion is difficult. A series of primitive controllers are used, concurrently combined and sequenced to produce two hierarchical controllers: the *Guarded Move Controller* and the *Compliant Insertion Controller*. The former generates a guarded approach that positions the tool at an optimum location for insertion. Upon reaching an appropriate insertion position the finite state machine moves to the next state initiating the latter controller, which drives an insertion. To do so successfully, misalignments are corrected to decrease friction and resolve jamming and wedging phenomena. Both of these controllers were used by an industrial robot; while, a modified version of these was also used by a dual-arm humanoid robot (the modified version accounted for the combined effect of two serial-link chains). Additionally, a third compound controller was designed for the hold-role. This controller seeks to emulate the rigid hold a human would enforce when a second person pushes a part

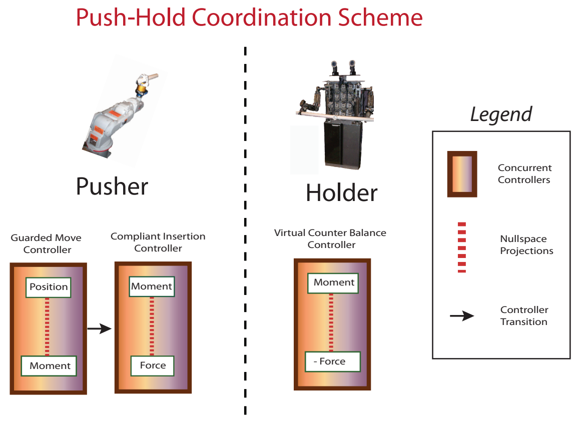


Fig. 1. The HP3JC inserts a male truss into a female counterpart held by ISAC. Pushing requires two compound controllers effected through the null space approach. Holding requires one compound controller.

to produce an insertion. The controller seeks to counter-act the experienced forces while simultaneously displacing the orientation of the part to facilitate the entry of the mating truss. An illustration of the push-hold coordination scheme can be seen in Fig. 1.

## A. Guarded Move Controller

The guarded move controller $\pi_{GM}$, uses a dominant position controller, $\phi_p$, and a subordinate moment residual controller, $\phi_{mr}$. The order is decided empirically by prioritizing the need for an optimal insertion location. The position controller displaces the rigidly held truss to such location, $(x_{ref})$, and a subordinate moment controller minimizes perturbations if contact is made during the trajectory. The position controller receives a 3D reference cartesian position from a stereo visual system that detects color fiducial marks placed at the fixture's tips [18].

For the position primitive, the sensor transform converts image coordinates to cartesian positions: $s_p(\gamma_{visual\_sys})$, while the effector transform maps the updated cartesian position to the robot's current joint configuration: $e_p(\gamma_{joint})$. On the other hand, the moment residual controller has a sensor transform, $s_{mr}(\gamma_{moment})$, that returns the moments experienced by the F/T sensor, and has an effector transform $e_{mr}(\gamma_{torque})$, that converts torque updates into joint angle updates. The composite guarded move controller, $\pi_{GM}$, is synthesized by having the moment controller be subject-to the position controller and defined as:

$$\pi_{GM} = \phi_{mr} \left|^{s_{mr}(\gamma_{moment})}_{e_{mr}(\gamma_{torque})} (m_{ref}) \triangleleft \phi_p \right|^{s_p(\gamma_{visual\_sys})}_{e_p(\gamma_{joint})} (x_{ref}). \tag{9}$$

## B. Compliant Insertion Controller

The compliant insertion controller minimizes residual moments and forces experienced during the assembly's insertion stage. As stated earlier, experimental practice suggests that the aligning of the truss takes precedence over its position during the insertion stage. Hence, the hierarchical controller is composed of a dominant moment residual primitive, $\phi_{mr}$, and a subordinate force residual controller, $\phi_{fr}$. The subordinate controller uses a force reference $(f_{ref})$ to generate the driving force to execute the insertion. The dominant moment residual controller aligns the truss as it is inserted into the fixture. The alignment is a function of the experienced forces produced as the truss collides with the fixture's interior wall. The compliant insertion controller is defined as:

$$\pi_{CI} = \phi_{fr} \left|^{s_{fr}(\gamma_{force})}_{e_{fr}(\gamma_{torque})} (f_{ref}) \triangleleft \phi_{mr} \right|^{s_{mr}(\gamma_{force})}_{e_{mr}(\gamma_{torque})} . \tag{10}$$

Screen-shots of the assembly demon are shown in Fig. 2.

## C. Counterbalance Controller

As part of the push-hold schemes, a composite controller was devised to implement a force guided system that would maintain the static fixture's position in place while updating its orientation. Following the design evidence of the compliant insertion controller, the counterbalance controller, $\pi_{CB}$, seeks to facilitate the insertion process and is composed of a dominant moment residual controller and a subordinate force controller:

$$\pi_{CB} = \phi_{fr} \left|^{s_{fr}(\gamma_{force})}_{e_{fr}(\gamma_{torque})} (-f_{ref}) \triangleleft \phi_{mr} \right|^{s_{mr}(\gamma_{force})}_{e_{mr}(\gamma_{torque})} \tag{11}$$

Though similar to the compliant insertion controller, $\pi_{CB}$, opposes the force applied by an incoming truss reference and displaces the end-effectors so as to create an optimal entry angle for the mating truss. In this way, it minimizes moment and force residuals throughout the task. A similar version of the three prior controllers was adapted to be used with ISAC [19].

## V. EXPERIMENTS

The demonstration seeks to achieve collaborative and cooperative assembly under a multi-agent distributed architecture. The latter also encapsulates the controllers introduced

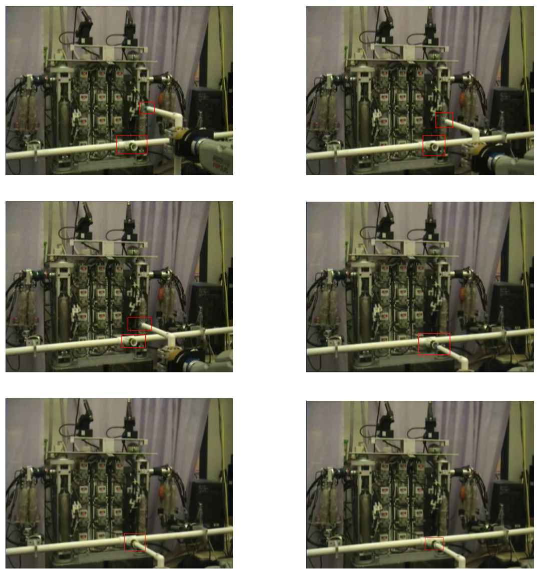


Fig. 2. Experiment 6: Two heterogeneous robots cooperate to perform a joint assembly using force sensing under a push-push coordination scheme.

in Sec. IV to implement parts insertion as outlined in Sec. IV-B. Before presenting the experimental set-up details, the hardware is described below.

The testbed consists of an: HP3JC with a JR3, six-axis F/T sensor, and a Barret Hand mounted on the wrist; and an in-house built humanoid robot, ISAC. The anthropomorph has two manipulators, each actuated by 12 pneumatic McKibben artificial muscles. Each of ISACs end-effectors consist of an ATI six-axis F/T sensor and a machined aluminum bracket specially designed to hold the truss. Truss', both male and female were made from commercial PVC piping. The male truss was composed of two 0.5 in. pipes connected by an elbow connector. At the truss' tool-tip an inverted chamfer was used to facilitate its entry into the female counterpart. The female truss was a 1.0 in. pipe connected through a t-connector to two 1.0 in. pipes that were held rigidly by ISAC's aluminum brackets. Additionally, color segmentation was used with empirically set parameters for a low-pass filter and morphological operations. ISAC used image processing to compute the cartesian coordinates of both trusses with very good results and little noise.

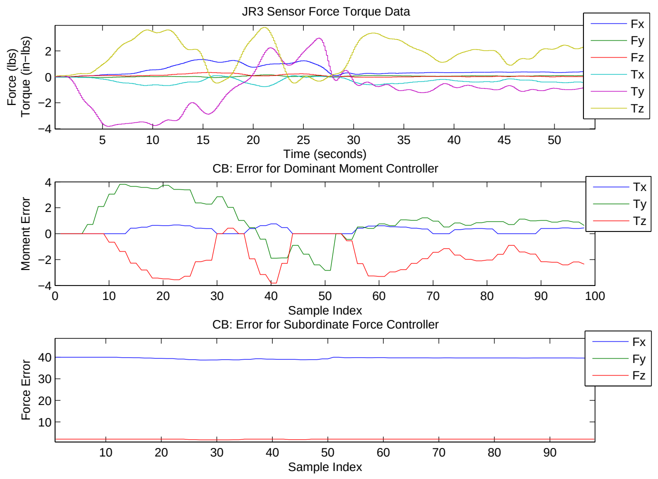
## A. Experiment 1: HP3JC-Push, ISAC-Hold

In the first demonstration, the industrial robot drove a male truss into a female fixture held by ISAC. The HP3JC robot used the $\pi_{GM}$ and the $\pi_{CI}$ controllers to actively perform the insertion, while ISAC used the $\pi_{CB}$ controller to counteract, but optimize entry forces exerted by the industrial robot. Note the reference parameter for the compliant insertion controller—a force reference—is what enacts the forward motion of the truss. In other words, this parameter affects the speed and force of the insertion. Force sensing, in this sense, drives the insertion, while adjusting the position of the truss in the vertical and horizontal planes.

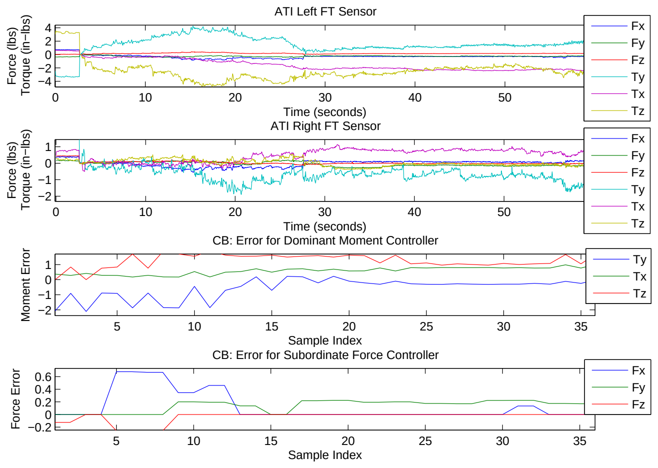
Additionally, three metrics were used to measure the assembly tasks' performance: (a) time-to-completion, (b) the sum of the absolute value of moment residuals in the x-, y-, and z-directions (referred to hereafter as "moment errors"), and (c) the reference force parameter. The latter was used to distinguish between faster and slower insertions driven by the industrial robot. Faster insertions used a force reference value of Fx=40 lbs and slower insertions used a value of Fx=20 lbs. Insertions were assumed complete after the male fiducial mark was covered.

Sensory data for one trial is presented for both robots in Fig. 3(a) and Fig. 3(b) in three sub-plots representing the: (i) force and torque signatures, (ii) the moment residuals, and (iii) the force residuals.

The data for this demo shows the assembly was completed in 54 seconds. The duration of this and other experiments may seem significant. The primary reason for such durations was the inability to access the industrial robot's low-level control loop. Such impediment forced us to operate through the industrial robot's API which prevents pre-emptive motion and significantly delayed the overall response of the system. With respect to moments, note that residual in the y-direction was reduced efficiently by both the HP3JC and ISAC, and



(a) Force data for the HP3JC-pushing robot



(b) Force data for the ISAC-holding robot.

Fig. 3. Exp 1: Force signatures for the HP3JC robot and ISAC.

moments in the z-direction maintained a small presence in both robots. The presence of moments in the z-direction indicate the approach by the truss also contained a horizontal displacement. Small residual moments in the y- and z-directions were expected given that there is a difference in diameters between the male and female, which allows the truss' to exert mutual load from the spacing between them. Additionally, for ISAC, the virtual counter balance controller seeks to eliminate force errors by adjusting the female fixture position. Forces in the x- and z-directions converged to zero. The y-direction residual force error is a response to the presence of moment in the z-direction from the male truss, which had not fully converged by the time the assembly was finalized.

A summary of metric results across trials is shown in Fig. 4. The first three trials were run under the slower force reference value Fx=20, which in general, yielded slower assemblies as opposed to the trials run with Fx=40. With respect to moment residuals, the HP3JC robot experienced moment errors greater than those experienced by ISAC. The last five experiment trials used the faster force reference

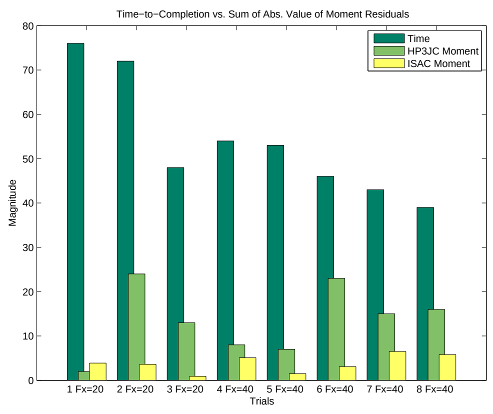


Fig. 4. Exp.1: Results summary across 8 trials.

value Fx=40. The higher force reference led to *generally* shorter completion times and hinted at larger moment errors for both robots than the ones registered in the first three trials. Two of the first three trials contained larger than normal moment errors due to one wedging in one trial and jamming in another.

## B. *Experiment 2: ISAC-Push and HP3JC-Hold*

The second demonstration reversed roles and assigned ISAC as the active robot in the insertion task. This experiment, as opposed to the previous one, began by having the male truss held by the HP3JC at a ready-position in front of the female fixture held by ISAC. The humanoid's virtual-contact compliant insertion controller, $\pi_{VCI}$ was responsible for driving the insertion. In this experiment, a force reference parameter of Fx=-0.5 lbs was used by the subordinate controller of the virtual compliant insertion controller for ISAC. The results are shown in Fig. 5.

Note that the moment residual errors are a result of the average moment's contribution from both the right and left F/T sensors. A slower speed was selected for the motion of the pneumatic actuators to achieve greater accuracy. From clock time 1–200 seconds, two important patterns are present in the data: a) there is a roughly constant increment in the moment residual error in the y-direction, and b) the force reference force is gradually canceled over this period. Both patterns indicate the existence of stiction in the task. This is further corroborated by the quick fall seen in the left torque reading in the y-direction and the quick increase in the force reference value Fx in the negative direction. The controllers are unable to decrease residuals caused by stiction until the corrective force generated by the controllers overcome the sticking forces present through the artificial muscles. After the sticking forces were overcome, the insertion took place quickly. The sudden completion of the task did not allow the compound controllers enough time to converge back to their reference goals.

For ISAC the averaged moment errors generated in this experiment were greater than the ones in experiment 1. ISAC experiences greater stress as it drives the insertion than it did when it used the counter balance controller. On the other hand, the averaged moment residual errors experienced by ISAC are lower than the ones experienced by the HP3JC robot. This suggests that the use of artificial muscles eases impact or stress induced as compared to rigid industrial robots.

A summary of the results across six trials is found in Fig. **??**. All trials in this experiment were run with the same force reference value. Three trials experienced longer times-to-completion due to more prominent stiction phenomena.

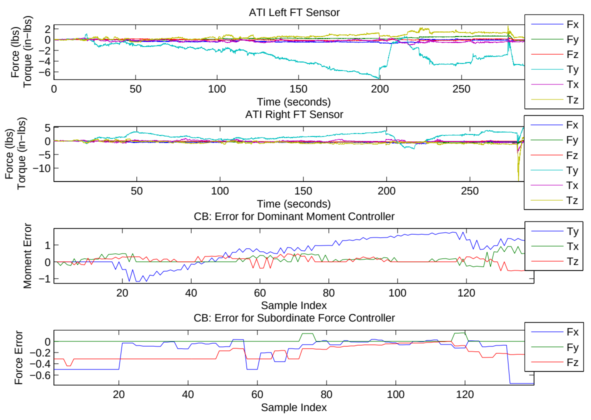


Fig. 5. Exp 2: Quick changes in torques due to stiction effects caused by ISAC's compliant nature.

## VI. Discussion

As experimental results showed, the collaborative assembly task across a heterogeneous multi-robot team was successfully tasked. The multi-agent architecture fulfilled its design paradigm and was able to encapsulate low-level and high-level abstractions. The middleware communication layer functioned well throughout the experiments and distributed a vast amount of visual and numeric data to different components in a decentralized manner. A number of finite state machines were run under different agent engine representations to automate ~~all aspects of~~ the robotic system.

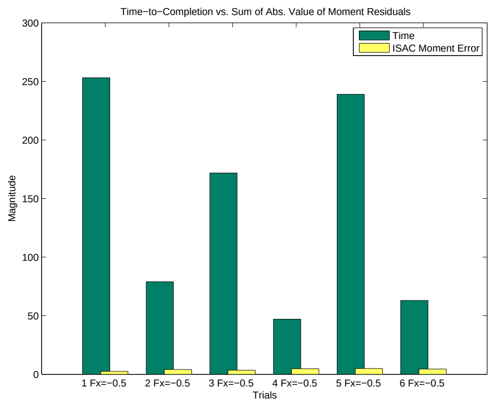


Fig. 6. Exp.2: Summary for results across 6 trials.

Additionally, the control basis framework allowed for the modular implementation of the basis controllers within the overall paradigm of our agent architecture. Basis controllers were successfully compounded and sequenced. The control policy ~~to enact the controllers~~ ~~was~~ successfully implemented at the highest level of abstraction allowing for a smooth transition of compound ~~control~~ basis controllers.

The overall system was flexible in that by selecting well defined control laws, a variety of controllers for manipulation, insertion, and counterbalancing were deployed in two very different robots with two very different physical characteristics. The results presented above show that the controllers reduce moment and force errors over time over the duration of the insertion task. Even in situations were disturbances like stiction were present (Exp. 2), the controllers were able to overcome such phenomena and successfully complete the assembly task.

While there positive aspects of this model have been presented, the author would also like to comment on the challenges posed by such a system as well. That is, ~~that~~ as the number of robots increase, and along with it, the number of sensory and hardware modules, so does the size of multi-agent architecture. While these architectures are designed to scale easily and facilitate integration, there is a ~~very~~ significant overhead needed to set them up. More development tools that can aid in the automatic initialization and resetting, and the switching on–or–off of debugging information would be very desirable.

## VII. Conclusion and Future Work

This paper presented a team of heterogeneous robots performing an autonomous and collaborative assembly task that is grounded on a distributed multi-agent architecture and modular control basis approach. The modularity and the flexibility of the agent-based architecture and the control approach worked in concert to bootstrap robust, flexible, and decidedly reactive controllers. Experimental results concluded that multi-agent multi-robotic collaborative systems with modular control approaches is a viable approach to generate complex robotic behavior.

For future work, the authors would like to extend the role-playing that robots play in assembly tasks, mimicking human behavior. On occasion, when two entities (humans, or two arms of a human) are trying to assemble an object, and the insertion is not easily accomplished, humans tend to try to push from both ends simultaneously. This kind of assembly is prone to higher forces, quicker motions, and error but humans practically choose such an approach in an attempt to assemble parts. Such a task would be an interesting test to examine the robustness and flexibility of our system.

## VIII. Acknowledgments

The authors gratefully acknowledge the contribution of National Research Organization and reviewers' comments.

## References

[1] T. Brogardh, "Present and future robot control development–an industrial perspective," *Journal of Annual Reviews in Control*, vol. 31, no. 1, pp. 69–79, 2007.

[2] J. Rojas and R. A. Peters II, "Preliminary results in force guided assembly for teams of heterogeneous robots," *SPIE's Defense, Security, and Sensing. Accepted and pending publication.*, April 2009.

[3] J. Kramer and M. Scheutz, "Development environments for autonomous mobile robots: A survey,," *Autonomous Robots*, vol. 22, no. 2, p. 101132, 2007.

[4] R. E. Olivares, "The intelligent machine architecture version 2.5: A revised development environment and software architecture," Master's thesis, Vanderbilt University, 2003.

[5] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. B. Foote, J. Leibs, R. Wheeler, , and A. Y. Ng, "Ros: an open-source robot operating system,," in *ICRA workshop on Open-Source Software*, 2009.

[6] M. Huber, "A hybrid architecture for adaptive robot control," Ph.D. Dissertation, University of Massachusetts, Sept. 2000.

[7] M. Namvar and F. Aghili, "Adaptive force control of robots in presence of uncertainty in environment," *In the Proceedings of the 2006 American Control Conference*, pp. 3253–3258, June 2006.

[8] P. Yuan, "An adaptive feedback scheduling algorithm for robot assembly and real-time control systems," *In the Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pp. 2226–2231, October 9-15 2006.

[9] J. Rojas and R. A. Peters II, "Sensory integration with articulated motion on a humanoid robot," *Journal of Applied Bionics and Biomechanics*, vol. 2, no. 3-4, pp. 171–178, 2005.

[10] R. T. Pack, "Ima: The intelligent machine architecture," Ph.D. Dissertation, Vanderbilt University, 2201 West End Avenue, Nashville, Tennessee 37235, May 1999.

[11] J. Jameson and L. Leifer, "Automatic grasping: An optimization approach," *Proceedings of the IEEE Transactions on Systems, Man, and Cybernetics*, vol. 17, no. 5, pp. 806–813, Sept. 1987.

[12] J. Son, R. Howe, and G. Hager, "Preliminary results on grasping with vision and touch," *In the Proceedings of the IEEE Conference on Intelligent Robots and Systems*, vol. 3, pp. 1068–1075, Nov. 1996.

[13] O. Brock, A. Fagg, R. Grupen, R. Platt, M. Rosenstein, and J. Sweeney, "A framework for learning and control in intelligent humanoid robots," *International Journal of Humanoid Robots*, vol. 2, no. 3, pp. 301–336, 2005.

[14] R. Platt, A. H. Fagg, and R. A. Grupen, "Nullsapce composition of control laws for grasping," *In the Proceedings of the IEEE International Conference on Intelligent Robots and Systems*, pp. 1717–1723, 2002.

[15] R. J. Platt, "Learning and generalizing control-based grasping and manipulations skills," Ph.D. dissertation, University of Massachusetts Amherst, 2006.

[16] J. A. Coelho and R. A. Grupen, "A control basis for learning multifingered grasps," *Journal of Robotic Systems*, vol. 14, no. 7, pp. 554 – 557, 1997.

[17] H. Inuoe, "Force feedback in precise assembly tasks," *In Artificial Intelligence: An MIT Perspecrive*, vol. 2, pp. 219–241, 1981.

[18] J. Rojas and R. A. P. II, "Preliminary results in force-guided assembly for teams of heterogeneous robots," in *Proceedings of the SPIE Conference on Defense and Security*, vol. 7332, no. 1, 2009.

[19] J. Rojas, "Autonomous cooperative assembly by force feedback using a control basis approach," Ph.D. dissertation, Vanderbilt University, 2009.