October 2007

Towards Robotic Self-reassembly After Explosion

Mark Yim
University of Pennsylvania, yim@grasp.cis.upenn.edu

Babak Shirmohammadi
University of Pennsylvania

Jimmy Sastra
University of Pennsylvania

Michael Park
University of Pennsylvania

Michael Dugan
University of Pennsylvania

See next page for additional authors

Recommended Citation

Yim, Mark; Shirmohammadi, Babak; Sastra, Jimmy; Park, Michael; Dugan, Michael; and Taylor, Camillo J., "Towards Robotic Self-reassembly After Explosion" (2007). *Departmental Papers (MEAM)*. 147.
http://repository.upenn.edu/meam_papers/147

Reprinted from *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2007*, October 2007.

Towards Robotic Self-reassembly After Explosion

Abstract
This paper introduces a new challenge problem: designing robotic systems to recover after disassembly from high-energy events and a first implemented solution of a simplified problem. It uses vision-based localization for self-reassembly. The control architecture for the various states of the robot, from fully-assembled to the modes for sequential docking, are explained and inter-module communication details for the robotic system are described.

Comments
Reprinted from *Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2007*, October 2007.

Author(s)
Mark Yim, Babak Shirmohammadi, Jimmy Sastra, Michael Park, Michael Dugan, and Camillo J. Taylor

Proceedings of the 2007 IEEE/RSJ International
Conference on Intelligent Robots and Systems
San Diego, CA, USA, Oct 29 - Nov 2, 2007

Towards Robotic Self-reassembly After Explosion

Mark Yim, Babak Shirmohammadi, Jimmy Sastra, Michael Park, Michael Dugan and C.J. Taylor

Abstract— **This paper introduces a new challenge problem: designing robotic systems to recover after disassembly from high-energy events and a first implemented solution of a simplified problem. It uses vision-based localization for self-reassembly. The control architecture for the various states of the robot, from fully-assembled to the modes for sequential docking, are explained and inter-module communication details for the robotic system are described.**

I. INTRODUCTION

The April 2007 special issue of IRAM [1] had a theme on grand challenges of robotics. It included the grand challenges from a variety of robotics specialties. One of the grand challenges proposed for modular self-reconfigurable robots is the ability for a system to repair itself after being exploded into many pieces. The effort to solve this grand challenge pushes the technical ability for integrated systems to plan and execute self-assembling hardware and software under unstructured conditions. Solving the challenge will show an unprecedented level of robustness in a robotic system. Robustness is one of the three promises of self-reconfiguring modular robotic systems [2], the others being versatility and low cost.

This paper introduces the problem, the issues involved, and one implementation towards this goal. The implementation demonstrates reconfiguration using a relatively small number of modules rather than the thousands of components ultimately envisioned. This paper is organized as follows: Section II presents the Robotic Self-reassembly after Explosion problem, its value, and some of the issues with references to existing work. Section III goes into some technical detail about the problem. Section IV presents an implementation towards solving the problem. Finally, Section V presents future work and conclusions.

II. ROBOTIC SELF-REASSEMBLY AFTER EXPLOSION (SAE)

The SAE problem, involves a system putting itself back together after being exploded. The main word to define is *explosion*. Explosion in this context is defined as the rapid randomized disassembly of a system from a high-energy event.

Grand challenges are often best described by what a demonstration of a solution would look like. For the SAE problem, the solution would show this sequence:

1) Doing a task.
2) Being exploded into many pieces.
3) Self-repair (self-assembly).
4) Resuming the original, pre-exploded task.

A. *Structured disassembly from an unstructured event*

One key aspect of the solution presented we call *structured disassembly.* We add structure to the explosion by designing the system to break along specified boundaries. Engineering solutions for structured disassembly ensures that the bonds between modules are the only bonds that will be broken in an explosion. Typically, this is done by designing the system such that the target bonds are weakest relative to the forces and torques seen during an explosive event (e.g., an impact).

Some may argue that rather than spend efforts to disassemble in a structured manner and then reassemble, efforts should be spent to make sure the system won't break into pieces in the first place. One response to this argument is that there may be unexpected conditions in which forces are larger than planned for, such as an earthquake or terrorist activity. Even beyond this, there are situations where breaking apart may be desired. Just as car bumpers are made to crumple to absorb the energy of an impact, the disassembly of specific bonds holding a structure together may also absorb the energy of an impact. Ski boot detachment devices are an example of a system where structured disassembly helps to protect more fragile components, such as injury to feet and legs. Here, an important metric in analyzing the level of recoverable explosion is the amount of energy absorbed by the breaking of bonds.

B. *Self-repair*

Robotic self-assembly falls under the larger umbrella of self-repair. Essentially, self-repair involves the repairing of a broken system, either with the system replacing faulty components with redundant ones, or by fixing broken ones in-situ. This paper includes self-repair by self-reassembly. Self-repair can be broken down into three steps: diagnosis, planning, and execution

Diagnosis: *Identify, sense that a problem exists, and determine the cause of failure*. Diagnosis requires some reasoning about cause and effect, understanding of the physical processes of the system, and possibly reasoning about data from the history of a variety of sensors [3]. In the SAE case, the most obvious failure mode is that system is in

---

Manuscript received April 9, 2007.

M. Yim, J. Sastra, M. Park and M. Dugan are with the Mechanical Engineering and Applied Mechanics Dept. B. Shirmohammadi and C.J. Taylor are with the Computer Information Science Dept at the University of Pennsylvania, Philadelphia, PA. (e-mail: {yim, jsastra, parkmich, mdugan}@ seas.upenn.edu {babaks, cjtaylor}@cis.upenn..edu).

pieces. In this paper, we will not consider other failure modes (e.g. electrical components or internal structures damaged from impact.). Diagnosis then involves determining the connectedness of all the pieces. After this, the sensed arrangement of pieces then feeds into developing a plan.

Planning: *For given classes of failure, determine a sequence of actions that will fix the problem*. Here repair is essentially reassembly and reduces to the classic AI assembly problem [4], except that the pieces move themselves rather than being moved by a robot arm. Another difference for modular robots is that since there are often many identical modules, there are many configurations that are isomorphic [5].

Execution: *Implement the plan*. Executing a repair typically involves multiple hierarchical closed loop control processes, including the removal or rearrangements of damaged parts. For the SAE problem, the parts are already separated, so the main objective is the motion of modules in the environment to dock with other modules.

C. Related work and metrics

Modular self-reconfiguring robot systems have achieved several of the elements described in the execution and planning phases of self-repair. Murata [6] demonstrated repair of many identical modules connected in one connected component. Chirikjian demonstrated robotic self-repair using Lego systems in the context of self-replication though the environment was structured [7].

Another element necessary for SAE is the relative localization of parts after explosion. As vision sensors and computing elements continue to get smaller, cheaper, and faster, it has become increasingly attractive to consider the use of smart camera networks. Each camera node has its own imaging device, processing unit, and communication unit in a self-contained package. Other approaches to recovering the relative positions of a set of cameras based on tracked objects have been proposed in the literature [8-12]. These approaches can be very effective in situations where one can gather sufficient correspondences over time. In contrast, the approach used here [13] directly instruments the sensors and provides rapid estimates of the sensor field configuration using modest computational and communication resources.

Docking mechanisms are important elements that have been studied for modular robots [14-17]. However, the other elements of this task are relatively new, especially with regard to the randomness in exploding apart the elements.

One of the metrics that could be used to define the "randomness" of a particular implementation of SAE would be the entropy or disorder of the system after explosion. For example, at one end of the spectrum, a system that was exploded into just two pieces that fell next to each other with out any significant rotational misalignment would have the minimum randomness SAE metric. Much farther along the spectrum would be a robot exploded into thousands of pieces that were randomly strewn over a large area.

In the modular self-reconfigurable robot community, this type of re-assembly would be categorized in the mobile class [18] of self-reconfiguration, as there are multiple connected components that must move in the environment that come together. The minimal randomness example above (a system assembling two pieces that are relatively close in alignment) was demonstrated by Shen with CONRO [16].

Murata in [19] showed a camera aided docking method similar to this work. However, that system used only one camera multiple LED's within a group of modules in a known shape which made it more vulnerable to occlusion and low resolution problems. The video was broadcast offboard for processing, whereas this system has local computation with multiple cameras (each cluster has one). This leads to better relative position and orientation estimates with better abilities to handle occlusion.

Also in the mobile class of self-reconfigurable robots, the Swarm robot [20] demonstrated linking together tens of mobile robots with small grippers. In this case, the robots are built to drive around and grab onto each other. This system focuses more on a group of robots without control for one connected component. Also, none of these demonstrations include any high-energy events (explosions) or randomized distribution which pushes on the robustness of the methods.

III. TECHNICAL ASPECTS

A. Structured disassembly

The context of explosion, as defined above, includes a high-energy event. The question is how high is high? Again, in this context, any event that injects enough energy to break the bonds holding a structure together is high enough to be called an explosion. By designing bonds between modules as the weakest bonds in a system, they will likely be the bonds which break first. If the inertial properties of the modules are also small, the modules' bonds will also likely be the only bonds to break even under larger energy events.

While the goal is to develop systems that can self-reassemble after a large impact, it is easier to start with systems that self-reassemble after small impacts. This is done by designing module bonds that are relatively weak, but not too weak. At a minimum, the bonds must be strong enough to maintain integrity during normal tasks (i.e. under gravity and the applied forces and torques from environmental interactions).

B. Self-assessment (finding location of parts)

In this process, the system must identify which pieces are detached and where the parts are located. This is primarily a sensing activity. There are two sensing modalities required: connectivity and relative location.

Two modules must be able to detect that they are connected (or not), both when they lose connectivity after an explosion and when they re-establish connectivity during reassembly.

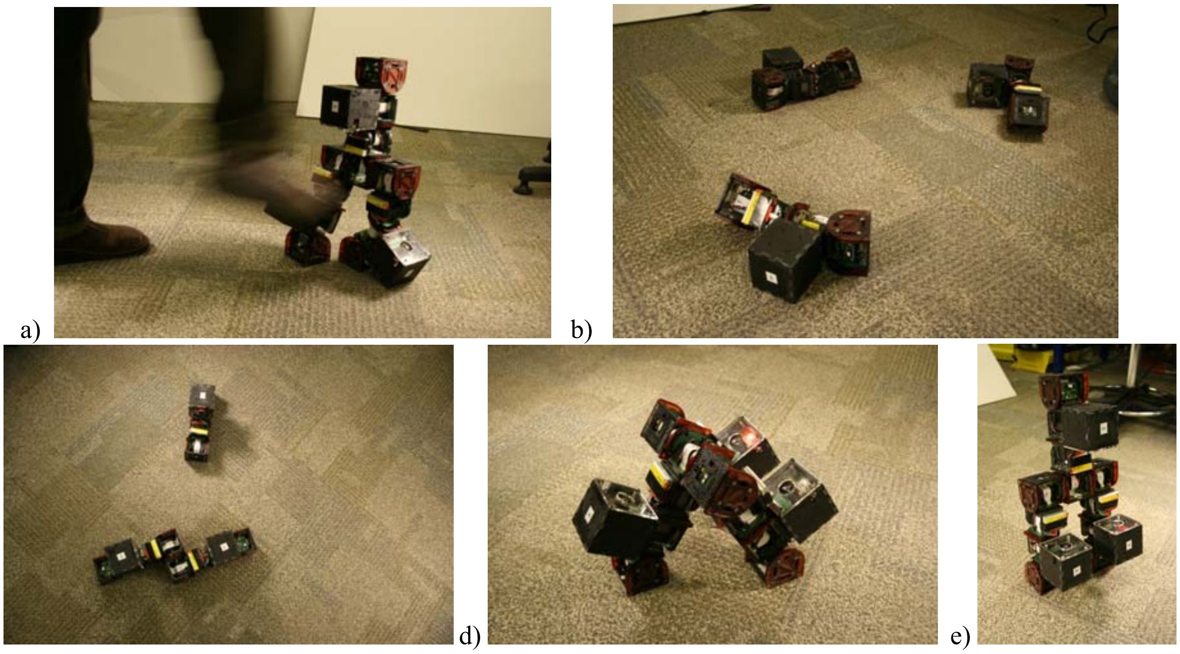

Fig. 1: Three piece self-reassembly after explosion. a) kick to midsection, b) resulting in three clusters of modules strewn randomly, c) clusters self-right and dock, d) system stands up, 3) system resumes walking.

The modules must also be able to find the relative positions and orientations of the other disconnected modules in order to re-dock with them. In our implementation, the camera nodes signal their presence by blinking their lights in a preset pattern. That is, each of the nodes would be assigned a unique string representing a fixed speed blink pattern such as 10110101. The node would then turn its light on for 1 and off for 0 in the sequence prescribed by its string. These blink patterns provide a means for each of the nodes to locate other nodes in their images. They do this by collecting a sequence of images over time and analyzing the image intensity arrays to locate pixels whose intensity varies in an appropriate manner.

This approach allows the camera node to both localize and uniquely identify neighboring nodes [13]. In addition, each camera node also has an integrated 3-axis accelerometer. This sensor allows each cluster to self-right itself into locomotive position (with the camera upright), if necessary. As soon as there is a line of sight between two camera nodes, they both can be localized up to a scale factor. The size of the blinking light in the image is a function of relative angle and distance between the two camera nodes. The LED size in the image is effective for determing distance at close range where accurate measurements are needed.

C. Planning

Planning occurs at two levels. At the higher level, the modules must plan for the connectivity of the assembled system. For modular robots, there are many repeated modules within the system. So, when an explosion occurs, the reassembly of the modules need not have the same modules in the same places as the original. An optimal plan for reassembly may involve minimizing the total distance and energy of all travel. In the event that some modules are damaged, the reassembly may move the damaged modules to locations which are not critical for operation, thereby increasing robustness.

At the lower level the moving modules must plan their collision free motion for docking. In the broadest sense, this becomes the standard robot motion planning problem, possibly in the presence of obstacles.

Architecturally, both the planning and self-assessment may be either centralized or decentralized. In most cases, optimality is easier to evaluate and implement in a centralized approach.

D. Bring parts together (guided locomotion)

Once an assembly plan has been established, the plans must be executed. This includes the locomotion of the modules to bring their connection faces in proximity, docking, and re-bonding of the modules together. Typically, these stages are closed loop actions using the sensors to guide the motion of the modules for docking.

IV. IMPLEMENTATION

A 15 module robot was used in a first demonstration of the SAE problem and shown in [21]. Five modules were grouped together in a “cluster”. Each cluster consists of four CKbot modules, each with one rotational degree-of-freedom

(DOF), and one camera module, that are all screwed together. Cluster-to-cluster connections are held together by magnets, which serve as the weaker boundaries for structured disassembly.

A. Demonstration

The sequence for the demonstration is shown in Figure 1. In this implementation, the designated task is bipedal walking. Figure 1a shows the modules mid-stride when the explosion event occurs – a kick to the midsection. The system falls apart at the magnet face boundaries into the three pieces.

The pieces are now randomly located in six dimensions (position and orientation), as shown in Figure 1b. In this state, a periodic local communication event through the connecting faces (infrared signal), does not get an acknowledgement indicating to the modules that the clusters are no longer connected.

Each cluster has the ability to individually move on the plane, which is something that individual modules cannot do. However, they only do so when the camera is upright. In situations where clusters are not in an orientation that can travel, accelerometers in the camera module detect the orientation of the cluster and this in turn causes the unit to perform maneuvers to self-right.

Once upright, two clusters perform a search to find each other visually. The two approach each other such that two side faces can attach together, as shown in Figure 1c. The magnet faces provide a mechanism where the modules need only locate themselves within approximately one centimeter to dock.

The two attached clusters then move as one unit searching for the third cluster which is also searching. The docking procedure is similar to the previous process.

When the three clusters are together the full system is assembled, but is now lying prone rather than standing up. The system recognizes its state (all connected and prone). It then performs a standing gait, as shown in Figure 1d. Once upright, the system of three clusters senses its overall state again using the accelerometers, so the robot resumes walking as in Figure 1e.

B. Architecture

The architecture for this implementation includes modular hardware as well as communication and control strategy. Since the hardware is hierarchical – modules form clusters, clusters form systems – the communication and control structure and naturally follows that architecture as well.

1) Modular hardware

a) CKbot Modules

CKbot (Connector Kinetic roBot) is the modular reconfigurable robot platform for this work. The kinematics are similar to many chain style reconfigurable modular robots [14-16]. Each module in the system consists of:

1) A laser cut plastic (ABS) body with a hobby servo actuator to control one rotational DOF.

2) A controller (PIC18F2680) and associated hardware for implementing a Controller Area Network (CAN) and neighbor-to-neighbor IR communications protocol.

3) Four connector faces that pass the communications bus and power bus with an option of attaching at 90° rotations.

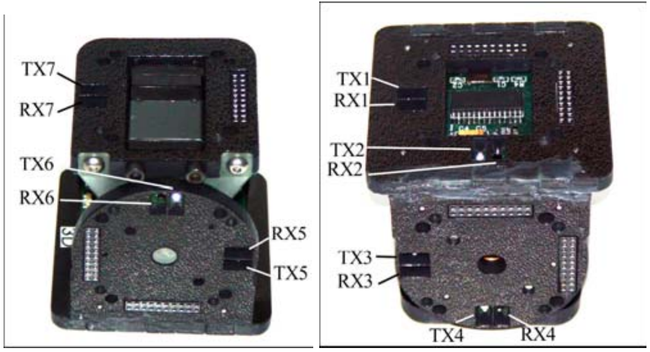

Fig. 2: Two IR transmitter and receiver pairs are on each side of a CKbot module except the bottom port which has one pair.

Fig. 3: One cluster of four CKbot modules with camera, controller, and magnetic face attachments.

One module can be viewed as a cube with connectors on top, bottom, left, and right faces as in Figure 2. The top, left and right faces are rigidly mounted together, the bottom face is actuated to rotate up to form the front or rear face of a perfect cube. Functionally the module has one symmetry where the module is rotated so that left and right sides are swapped. Figure 2 shows the layout of the seven IR pairs. Note that when two faces are attached together, the transmitter LED (TX) faces directly on to the receiver photodiode (RX) on the opposing face and vice versa.

Currently, about 60 CKbot modules have been constructed and a variety of tasks have been demonstrated including moving like a snake, dynamic rolling [22], digging in sand and walking like a slinky toy; see [23] for videos.

b) Camera module

The camera module is the cube with a window sitting on top of the four CKbot modules in Figure 3. Each camera module contains an SBC50 Camera by Vision Component with a fisheye lens that communicates through RS232 with a daughter board having a PIC18F2680 microcontroller, a 3-axis accelerometer, a wide angle LED and a Bluetooth module. Image processing is performed on the camera using the Vision Components Lib Image Processing Library. The daughter board provides a CAN interface for communication between the regular modules and the camera module. The two camera modules can communicate through Bluetooth or with different blinking patterns, though bluetooth was not used in the demonstration in this paper.

c) Magnet faces

Two modules are attached together using screws or optional magnet faces that physically connect two modules together. Magnet faces are screwed onto the sides of the modules.. These faces have 8 rare earth magnets with 4 north facing and 4 south facing magnets arranged such that two opposing faces will attract each other at 90 degree rotations. The faces have enough strength to hold 7 modules vertically before the weight of those 7 pulls them apart.

When modules are screwed together, an electrical header is included between modules to facilitate the CANbus and power bus. With magnet face connections, only IR data is transmitted and received between modules. Power can be supplied either from an external power supply or onboard Li-poly batteries that plug into the power ports on the module. Each modular cluster connected with magnets requires at least one source of power to interact with the other clusters.

The CANbus is global allowing fast module to module communications discovery within a cluster. The IR local bus enables module connectivity and communication between magnetically connected clusters of modules.

2) Communication and coordination

The inter-module communication structure is based on the Robotics Bus [24] which uses CANbus. CAN is used to coordinate communication within each cluster. In this situation, all components (modules, camera/accelerometer, controller) can communicate with one another with designated node IDs and message identifiers.

A serial infra-red (IR) communication method is also used for neighboring modules to communicate through their attached docking face. The controller processor inputs data from the camera/accelerometer and IR ports of the modules to determine what task to perform with the connected modules (e.g., self-right, move and turn toward another cluster of modules, determine inter-cluster connectivity).

When clusters of modules are connected at the specified docking points, the controllers of the two clusters communicate with an IR/CAN combination that allows them to know when and how two clusters are connected. For this work, one controller acts as a master to syncrhonize motions.of connected clusters.

3) Software

a) State machine

Each cluster must be able to take on different roles and perform different actions. If it is not connected to other clusters it must locomote on its own to find and connect to other clusters based on inputs from the camera module. If it is connected in a system of clusters, the master must send messages to the other clusters. The decision making can be described by the following state machine.

Connectivity: The controller sends messages on its IR ports to determine cluster-to-cluster connections. If it is connected to other clusters, only one of the clusters will become the master and command the other clusters.

Search: The cluster(s) rotates in place searching for other clusters. It knows it has found another cluster if the camera module sees the blinking LED pattern of another camera module. If the pattern belongs to a cluster it wants to dock to, it will enter the “approach and dock” state.

Approach and Dock: The camera module guides docking. Once docking occurs, the controller will enter the “connectivity” state to verify that docking was successful. This is facilitated with IR signal communication.

Walk: In this case the task of the full system of clusters is walking. If the system has enough clusters, the system will enter “walk” state. Here, the body decides the gait to be played, such as standing up, taking a left step or right step and sends this decision through IR/CAN to the other leg clusters. The controllers in the leg clusters wait and listen to gaits being sent through IR.

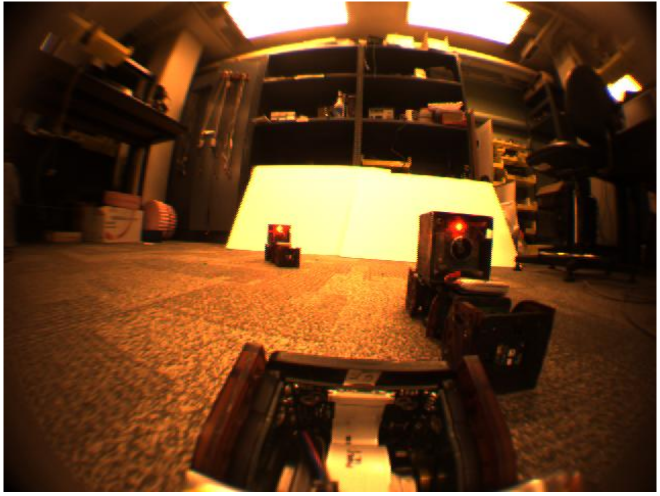

Fig 4: A view of the two other legs from the torso camera module. The wide angle fish-eye lens covers almost 120 degree.

b) Vision localization

The localization software consists of two parts: the first part runs directly on the camera and captures 16 images (as seen in Figure 4) at 20 fps, 2 times faster than the blinker rate, and looks for the 8-bit blinker patterns on odd and even frames; therefore, the blinking light will be detected on odd, even or both frame sets without any synchronization process. We assume each camera has unique IDs therefore it is not possible to have more than one connected component blinking blob with the same ID on a frame set. The output of the camera is the centroid, ID and size of the blinking blob. The second part of the software runs on the daughter board and determines the relative position and orientation of the nodes in 3D based on these images measurements and the accelerometer readings. The resulting pose information is relayed to the other modules using the Robotics Bus interface.

c) Posable programming

In order to create the various gaits for locomotion, a GUI that records the angular positions of the modules was used. The programmer manually shapes the robot into a desired position and records the joint angles. A series of these recorded positions then form a gait table which can be executed through the same GUI. This provides a very efficient way of designing and testing gaits, especially for large configurations. One of the advantages is that the gait can be designed to be stable by simply balancing the robot at every step. Another advantage of the GUI is that the gait table can be edited after it is recorded in order to tweak the position of a single module.

C. Results

Many experiments were performed as the software was being developed. It is estimated that 40 trials were performed with approximately 5-7 successful runs (where successful is defined to be achieving sequence above with no interference or with minor interference such as a random reorientation of a cluster.) Times for successful runs averaged between 6 and 7 minutes.

V. FUTURE WORK AND CONCLUSIONS

While three pieces is a modest number for an SAE demonstration, it contains all the components of required of SAE. Increasing the number of pieces (and thus the disorder metric) for the SAE problem is a clear goal.

There are several steps to increasing the disorder of the system. Near term work includes developing architectures where individual modules with limited functionality (no cameras, perhaps no ability to move) can be viewed and assembled by other clusters that do contain localization and mobility.

Besides increasing the number of pieces, improving assembly structured disassembly may also be useful. This includes demonstrating the ability to reassemble different isomorphic configurations [5] as well as developing a methodology for designing systems with breakable bonds

This paper introduces the SAE problem, which requires the integration of self-assembly, self-diagnosis, and hierarchical distributed control. While many of the individual components of solutions to the SAE problem are straight forward, the integration of all of them make this an excellent challenge problem to gauge the maturity of self-reconfigurable systems.

REFERENCES

[1] M. Yim, W-M Shen, B. Salemi, D. Rus, M. Moll, H. Lipson, E. Klavins and G.S. Chirikjian, “Modular Self-Reconfigurable Robot Systems [Grand Challenges of Robotics]” *IEEE Robotics and Automation Magazine* v. 14,1, pp. 43-52, Mar. 2007

[2] M. Yim, D. Duff, and K. Roufas, “Polybot: a modular reconfigurable robot,” in *Proc. of the IEEE Intl. Conf. on Robotics and Automation,* Apr. 2000, pp. 514-520

[3] J. deKleer and O. Raiman, “How to diagnose with very little information,**”** in *Proc. of Fourth Intl. Wkshop on Principles of Diagnosis*, 1993, pp. 160-165

[4] J.C. Latombe, *Robot Motion Planning,* Boston, Kluwer Academic Publishers, 1991.

[5] M.Park, S.Chitta, A.Teichman and M.Yim, “Automatic Configuration Recognition Methods in Modular Robots,” *Intl. J. of Robotics Research*, submitted , 2006

[6] S. Murata, H. Kurokawa, S.Kokaji, “Self-assembling machine,” *Proc. IEEE Int. Conf. on Robotics and Automation,*,May 1994. pp. 441-448

[7] W. Park, D. Albright, C. Eddleston, W.K. Won, K. Lee and G.S. Chirikjian, “Robotic Self-Repair in a Semi-Structured Environment,” in *Workshop Proceedings of Self-Sustaining Robotic Systems,* 2004

[8] B. Dungan, A. Rahimi and T Darrell, “Simultaneous calibration and tracking with a network of non-overlapping sensors,” in *Proc. IEEE Conf. on Comp. Vision and Patt. Recog*., 2004.

[9] J. Campbell, P.B. Gibbons, S. Nath, P. Pillai, S. Seshan, and R. Sukthankar, “Irisnet: an internet-scale architecture for multimedia sensors,” in *Proc. of the 13th annual ACM Intl. Conf. on Multimedia,* pp. 81–88, , USA, 2005..

[10] Wu chi Feng, et.al. “Panoptes: scalable low-power video sensor networking technologies,” in *Proc. of the 11th ACM Intl. Conf. on Multimedia*, pp. 562–571, 2003.

[11] C. H. Lin, T. Lv, and I. B. Ozer W.Wolf, “A peer-to-peer architecture for distributed real-time gesture recognition,” in *Intl. Conf. on Multimedia and Exposition,* 2004.

[12] W. Matusik, W. Buehler, R. Raskarand, L. McMillan, and S. J. Gortler, ”Image-based visual hulls.” in *SIGGRAPH*, 2000.

[13] C.J. Taylor , B. Shirmohammadi, “Self Localizing Smart Camera Networks and their Applications to 3D Modeling,” Intl. Conf. on Distributed Smart Cameras (ICDSC-06) , 2006

[14] M.Yim,Y. Zhang,K.Roufas,D.Duff, C.Eldershaw, “Connecting and disconnecting for chain self-reconfiguration with PolyBot,” *Mechatronics, IEEE/ASME Trans*. v 7:4, pp. 442-451, Dec 2002

[15] S. Murata, E. Yoshida, K. Tomita, H. Kurokawa, A. Kamimura,S. Kokaji, “Hardware design of modular robotic system,” in, *Proc. of IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, (IROS). pp. 2210-2217, Nov. 2000

[16] W-M. Shen, P.Will, “Docking in self-reconfigurable robots,” in, *Proc. of IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, (IROS). pp. 1049-1054, Nov. 2001

[17] M. Nillson, “Heavy-duty connectors for self-reconfiguring robots,” in: *Proc. of IEEE Intl. Conf. on Robotics and Automation,* (ICRA), pp. 4071- 4076, 2002

[18] A. Casal and M. Yim, “Self-reconfiguration planning for a class of modular robots,” in *Proc. SPIE Sensor Fusion and Decentralized Control in Robotic Systems II* Vol. 3839, pp. 246-257, 1999

[19] S. Murata, K. Kakomura, and H. Kurokawa “Docking Experiments of a Modular Robot by Visual Feedback,” *Proc. of IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*, (IROS). pp: 625-630,. 2006

[20] F. Mondada, L.M. Gambardella, D. Floreano, S. Nolfi, J.-L. Deneuborg and M. Dorigo, “The cooperation of swarm-bots,” in: *IEEE Robotics & Automation Magazine*, vol 12:2, pp 21-28, June 2005

[21] M.Yim, et. al. “Robustness and self-repair in modular robots,” *Video proc. of IROS 2007*

[22] J. Sastra, S. Chitta and M. Yim, “Dynamic Rolling for a Modular Loop Robot,” *Intl. J. of Robotics Research*, submitted , 2007

[23] Videos of ckbot, available http://modlab.seas.upenn.edu

[24] D. Gomez-Ibanez, E. Stump, B. Grocholsky, V. Kumar, and C. J. Taylor. “The robotics bus: A local communications bus for robots.” *Proc. of Society of Photo-Optical Instrumentation Engineers*, 2004