Structure and Markings as Stimuli for Autonomous Construction

M. Allwright, N. Bhalla, and M. Dorigo

IRIDIA – Technical Report Series
TR/IRIDIA/2016-008
December 2016
Structure and Markings as Stimuli for Autonomous Construction

Michael Allwright
Department of Computer Science
University of Paderborn
Paderborn, Germany
Email: michael.allwright@upb.de

Navneet Bhalla
Department of Chemistry and Chemical Biology
Harvard University
Cambridge, MA 02138, USA
Email: nbhalla@gmwgroup.harvard.edu

Marco Dorigo
IRIDIA
Université Libre de Bruxelles
Brussels, Belgium
Email: mdorigo@ulb.ac.be

Abstract—We present the implementation of our swarm robotics construction system (SRoCS), which uses partially-completed structures, as well as markings, to coordinate construction. SRoCS consists of autonomous robots and stigmergic building blocks, which we have designed for this project. The autonomous robots are a significantly upgraded version of the BeBot, a mobile robotics platform, which we have equipped with a specialized manipulator module to pick-and-place the stigmergic building blocks. Using a single robot, we demonstrate how structure and markings can be used to regulate construction. This paper represents a milestone in our research towards the realization of a swarm robotics construction system, which aims to be capable of building a variety of structures using multiple robots.

I. INTRODUCTION

Our swarm robotics construction system (SRoCS) is a research tool that we have designed to study how autonomous construction behavior can be regulated using structure and markings as stimuli. SRoCS consists of small autonomous robots and stigmergic building blocks, which we have designed specifically for this project. The small autonomous robots are a significantly upgraded version of the BeBot [1], a mobile robotics platform that we have also equipped with a specialized manipulator module to pick-and-place the stigmergic building blocks.

In this paper, we verify the functionality of our system by using a single robot to perform two tasks involving the manipulation of blocks with respect to different types of stimuli in the immediate environment. Furthermore, we present a demonstration of a single robot constructing a staircase. In this demonstration, the robot modifies the structure of a partially-built staircase, by adding marked blocks to it. In effect, these modifications to the structure allow the robot to communicate indirectly with itself, through the environment, regulating the construction activity.

This mechanism of indirect communication is called stigmergy and is based on observational research into the collective behavior of social insects such as ants, bees, termites, and wasps [2]. Social insects communicate indirectly through modifications of the environment. These modifications of the environment become stimuli, which influence the behavior of other insects. This feedback mechanism enables the regulation of various collective activities, including nest construction.

Our approach to autonomous construction is inspired by Theraulaz and Bonabeau. In their work, simulated agents moved randomly around a three-dimensional lattice, depositing bricks when a given substructure in the immediate environment was detected [3]. Since the depositing of bricks modified the simulated environment, this behavior triggered a feedback mechanism, further regulating the construction, leading to the formation of several insect-nest-like structures. The significance of this approach was reinforced by recent research into the construction behavior of the termite Procor- nitermes araujoii. This observational research revealed that structure is a stronger stimulus than volatile marking [4].

In our previous paper, we have introduced the concept of SRoCS prior to the implementation [5]. The work presented in this paper represents a milestone in our research towards the realization of a swarm robotics construction system, which aims to be capable of building a variety of structures using multiple robots. In contrast to centralized multi-robot systems, our system is decentralized and robot localization is restricted to what can be sensed in the immediate environment. While these restrictions might reduce performance, Beckers et al. have noted in their work on a decentralized multi-robot system for clustering, that the resulting system was more robust and inherently scalable [6].

II. BACKGROUND

Recent work in multi-robot construction successfully demonstrated various systems capable of building structures using both aerial and ground-based robots [7]–[11]. All of these systems, however, relied on centralized infrastructure for localization or to facilitate robot-to-robot communication. The applications of these systems are, therefore, limited to scenarios where we can guarantee both correct and continuous operation of the centralized infrastructure.

Decentralized multi-robot construction was also demonstrated by Werfel et al. [12], [13]. Although this work was inspired by termites, our analysis of the implemented behavior reveals that the robots localize themselves on a complete map of the target structure, which was generated offline. Although successful, this approach does not benefit from the robustness and adaptability, which is characteristic of systems based on social insect behavior [2].

Sugawara and Doi demonstrated the first swarm robotics construction system, capable of building two-dimensional
structures in simulation and on real hardware [14], [15]. The system was based on a variant of Deneubourg’s clustering model [16], where, due to limited sensing, the robots were unable to perceive the local density of building materials, or differentiate between building materials and a partially-built structure. To compensate for this, Sugawara and Doi used semi-active building materials to regulate the formation of structures.

III. SYSTEM ARCHITECTURE

Autonomous construction is a challenging task. In our research, we simplify the actuation and sensing aspects of this task, so that we can focus on the coordination aspects of swarm construction behavior. Our system significantly extends the BeBot mobile robot platform [1] using a combination of off-the-shelf components and 3D printed parts. In contrast to the decentralized multi-robot system proposed by Werfel et al. [12], [13], our mobile robots are not confined to a track and can roam freely about the environment. Our extensions to the BeBot included developing a specialized manipulator module, and upgrading the electronics so that the signal processing can be implemented on board. The manipulator module is used to pick-and-place the stigmergic building blocks that we have designed for this project. The stigmergic building blocks are semi-active construction materials that can be marked by the robots during construction. The blocks are cubes with a 55 millimeter side length and spherical magnets in the corners. These magnets give the blocks a self-aligning and reinforcing characteristic, reducing cumulative error, and increasing the strength of the containing structure. We placed a localizable tag on each face of the blocks to enable the computer vision to be implemented on a cell-phone grade microprocessor. Similar to the semi-active blocks proposed by Sugawara and Doi, our blocks contain a microcontroller and are capable of routing messages between blocks and through a structure using the wireless interface embedded in the faces of the blocks. The blocks also contain four multi-color LEDs on each face to facilitate one-way, block-to-block communication using a camera. The manipulator module operates like a forklift mast, with the exception that the blocks are attached/detached from the top face using four semi-permanent electromagnets. The manipulator module contains a camera module for tracking the blocks and is capable of creating structures up to a height of three.

IV. THE STIGMERGIC BUILDING BLOCK

A. Electronics

At the core of the stigmergic building block is a microcontroller, which executes the blocks software. As shown in Fig. 1, we have mounted the microcontroller on the central circuit board to manage power and the routing of data between the various interfaces. We have also provided a connector on the central circuit board for a Zigbee-based communication module, enabling debugging and remote monitoring. Each face of the block contains an additional circuit board with a near field communication (NFC) transceiver and a LED driver. The NFC transceiver allows messages to be sent or received, simultaneously on any face, enabling robot-to-block and block-to-block communication. Each LED driver controls the brightness of four multi-color LEDs on the respective face of the block. The robot can sense the color of these LEDs from a distance while inspecting the containing structure.

B. Mechanical Design

We assemble a block using seven circuit boards, a central board, and a board in each of the six faces. Four of these boards attach to the side face ports around the perimeter of the central board, while two, the top and bottom boards, connect via a cable to the top and bottom ports shown in Fig. 1. We print the chassis of a block using selective laser sintering (SLS). The chassis consists of four side covers, a top cover, and a bottom cover. The four side covers clip on to the circuit boards, which are attached to the perimeter of the central board. The top and bottom covers contain small containers in the corners, where we have inserted a six millimeter spherical magnet. These covers clip on to the two remaining circuit boards and slide over the four side covers, to complete the assembly of a block.

C. Software

We have written the software for the block in C++. After power on, a block initializes itself and waits indefinitely for an NFC message to be received on any of the six faces. On receiving a message, a block sets all the colors of all the LEDs to the value given in that message. A valid message contains a single character matching ‘0’ through to ‘4’, where ‘0’ switches the LEDs off, and ‘1’ through to ‘4’ selects a color calibrated to the four quadrants of the UV color space. We refer to these four colors as Q1, Q2, Q3, and Q4 throughout the rest of this paper. We have selected these four colors with respect to the UV color space to speed up image processing on the robot. This speed up occurs as the frame buffers for the UV color channels are directly available for processing, without requiring a prior color space conversion.
V. THE AUTONOMOUS ROBOT

A. Electronics

The original BeBot hardware consists of twelve equally-spaced range finders mounted to a molded interconnect device (MID) chassis. The range finders connect to a microcontroller on the MID chassis, which samples the sensors and provides access to the readings via a serial interface. Two motors mounted to the chassis form a differential drive, allowing the robot to move around the environment. Two circuit boards slot into the MID chassis and are responsible for routing the power, expansion port signals, as well as the sensor and actuator signals to a central microprocessor.

The microprocessor used in the original design is inadequate with respect to our computer vision requirements. We have therefore redesigned the two circuit boards around a later generation microprocessor. To reduce development time and manufacturing costs, we have used a Duovero Computer-on-Module (COM) from Gumstix. The COM includes a dual-core, multimedia microprocessor from Texas Instruments, which is clocked at one gigahertz and connected to one gigabyte of memory. Gumstix provides two variants of the COM, allowing for optional wireless and Bluetooth connectivity on our robots. We have routed the camera serial interface (CSI) of the microprocessor out to two camera module ports, allowing for simultaneous capture of video from two sources. We have designed a camera module that provides power management, as well as clock and reset signals for a 5MP OmniVision image sensor. As shown in Fig. 2, we have positioned four white LEDs around the image sensor to control the illumination of the captured scene. In contrast to using a USB camera, this approach enables the use of hardware-accelerated capture, resizing, and compression of the video stream. Performing these operations on the CPU would have used significant processing time, which can now be allocated instead to our computer vision requirements. We have also provided an SD card reader for recording the video stream from the cameras, a slot for a low-power wireless module, and power management for the COM. A single USB port connects to an integrated USB hub, providing a serial connection to the console of the microprocessor, as well as an Ethernet connection via USB On-The-Go (OTG). The hub is compliant with the USB battery charging specification, allowing for recharging of the robot’s batteries.

In addition to the upgraded circuit boards, we have designed a specialized manipulator module, which uses a stepper motor to control the vertical position of an end-effector, consisting of four semi-permanent electromagnets as shown in Fig. 3. We use these electromagnets to allow the robot to pick-and-place blocks into a structure. The permanent magnetic field of the electromagnets is sufficient to hold a block in place during transport. To enable the attachment and detachment of a block, we precharge four 6.8 millifarad capacitors to 25 volts, before routing the stored charge to the electromagnets. Depending on the direction of flow, the current either strengthens or weakens the magnetic field of the electromagnets, causing a block to attach or detach respectively. As shown in Fig. 2, we have also equipped the end-effector with four range finders (two mounted on the lower circuit board, one in front, and one hidden underneath) and a camera module tilted 45 degrees towards the ground. When the end-effector is at the maximum height from the ground (3.5 blocks, or 19.25 centimeters), the camera can see blocks on the ground up to approximately 35 centimeters away from the center of the robot. At the minimum height from the ground (1 block, or 5.5 centimeters), the blocks can be tracked until they disappear underneath the end-effector. At this point, the software can make use of the range finders to do final adjustments to the alignment, before picking up or placing a block. The NFC antenna for writing messages to a block is traced around a circular cut out on the lower circuit board.

B. Mechanical Design

We have printed the structural components of the manipulator module and the end-effector from a photopolymer resin using stereolithography (SLA) on a Form 1+ 3D Printer. The 3D render in Fig. 3 shows the mechanical aspects of the manipulator module. The module is approximately 30 centimeters tall, and once mounted on top of the BeBot chassis, gives the robot an overall height of approximately 37 centimeters. As shown in Fig. 4, we have connected a stepper motor to a worm gear, which interfaces a pinion to provide rotation to the lower shaft. Two sprockets rotate with the lower shaft to raise/lower the chains, which we have connected to the end-effector as shown in Fig. 3. The weight of the end-effector is offset using lead counterweights.

C. Software

We have distributed the software running on the robot over four microcontrollers and the main microprocessor. The first microcontroller is built into the MID chassis and provides readings from the chassis range finders over a
serial interface. Two microcontrollers are located on the lower circuit board of the BeBot and are responsible for USB connectivity, power management, sensor sampling, and controlling the drive system. The final microcontroller is located in the manipulator module and provides an interface to the NFC transceiver, the end-effector position controller, the electromagnet precharge circuitry, and the range finders located on the end-effector. The microprocessor runs a custom distribution of Linux. Driver modules loaded into the kernel make the sensors and actuators of the robot available to our high-level software. We have implemented the software on the microprocessor as a single executable, which controls all aspects of the robot’s behavior.

Our software consists of an image processing pipeline, the block tracking and structure detection algorithms, and a finite state machine. The image processing pipeline is the most computationally intensive part of our software and determines the update period. The pipeline performs the following operations sequentially: capture, tag detection, save, and stream (via the wireless network). The last three operations are optional, and with the streaming disabled, we are able to achieve an update period between 150 and 200 milliseconds. To keep the update period constant, we add a delay as required to force the period to 200 milliseconds.

The tag detection operation in the pipeline uses the AprilTag algorithm [17] to detect the tags on the stigmergic building blocks. We use the output of the detection operation with OpenCV to determine the tag’s three-dimensional pose, as well as the color of the LEDs around it. We then offset the pose into the middle of the block and normalize the rotation around each axis to ±45 degrees. We cluster the tags that belong to the same block by comparing the distance between the offset tags with a threshold. To track the change in location of the blocks between frames, we use the Hungarian algorithm with a modified cost matrix to accommodate new and lost blocks [18]. Finally, we cluster the targets (tracked blocks) into structures by comparing the distance between two targets with a threshold.

After our software has completed the tag, block, target, and structure detection, and has sampled the remaining sensors, it steps the finite state machine. A simplified state chart of the behavior is shown in Fig. 5. The robot starts by performing a spot turn in search of an unused block (we are not using random walk, as we have not yet implemented obstacle avoidance). When the robot detects an unused block, it approaches and picks the block up. If a target is lost or the alignment is out of range, the robot reverses until it has reacquired the target, before reattempting the approach. After the robot has picked up an unused block, it performs another spot turn until it finds a target matching the specified criteria. We can define the criteria in terms of the markings (Q1-Q4 LED colors) and in terms of structural aspects. Structure aspects can include the number of adjacent blocks, the detected size of a structure, or the height of a block from the ground. Once the robot finds a target matching the specified criteria, the robot approaches it and places the unused block relative to that target. Placement depends on
the matching criteria, and can be set to stack the block on top of a target, or place it against one of the sides of a target. The robot then reverses before performing a spot turn, in search of another unused block. After our software has stepped the finite state machine, it performs actuation before starting the next iteration of the control loop.

VI. SYSTEM VERIFICATION

To verify our system, we have designed two tasks to test the hardware and demonstrate how the robot can use both structure and markings as a stimulus. In the first task, we test whether the robot can perform a given action in response to a markings-based stimulus. As shown in Fig. 6, the robot must pick up an unused block and place it on top of another block that is located in a structure with the LEDs illuminated (Q1). The second task in Fig. 7 tests whether the robot can respond to a structure-based stimulus. In this task, the robot must pick up an unused block and place it against the larger of two structures. The robot estimates the size of a structure using a simple heuristic: If a block in the structure has more adjacent blocks than the currently selected block, select the other block, else use the number of adjacent blocks of the currently selected block as an estimate of the structure size.

For the robot to complete these tasks successfully, there must be no issues in the robot-to-block communication, the precharging circuitry for the electromagnets, the block tracking and structure detection algorithms, the finite state machine, and the control loop responsible for regulating the position of the end-effector. For both tasks above, failure to pick up the unused block, or to place it at the specified location, constitutes a failed trial. We ran 15 trials for each task and were able to obtain a success rate of 73.3% for the markings-based task, and 80% for the structure-based task. The distribution of the task run time for the successful trials is shown in Fig. 8. The primary failure mode was due to issues with the mounting of the wheels and motors on the BeBot platform. These issues cause uneven friction and occasional jamming, leading to incorrect placement of a block due to the misalignment between the robot and the structure. The issue with the drive system aside, the two tasks have verified that the rest of our system is functioning correctly.

VII. DEMONSTRATION

To show how our system can utilize structure to regulate construction we have designed a controller to build a staircase consisting of three columns. In addition to structure, the controller also utilizes the NFC transceiver to set the color of the LEDs on each block, storing information regarding the target height of each column in the environment. A description of the staircase is as follows: the highest column of the staircase has a height of three and is illuminated with the Q3 color; the middle column has a height of two and is illuminated with the Q2 color; the lowest column consists of a single block and is illuminated with the Q1 color. Extending the software as described in Section V-C, we modify the search for target state to select a target that matches the following criteria: The target is illuminated with
one of the Q1-Q3 colors, and the target is the highest from the ground in the frontmost column with respect to the robot. The robot inspects the structure and determines the color and height of the highest block on the frontmost column. If the height is less than that specified by the column color, the robot sets the color of the block to the same color as the current column and stacks the block on top. If the height is equal to that specified by the column, the robot sets the color of the block to that of the next column and places the block so that it extends the structure and starts a new column.

Fig. 9 shows the various stages of the robot constructing the staircase, a video of this demonstration is also included in the supplementary materials. Due to the issues with the drive system, we have assembled this demonstration from multiple runs.

VIII. CONCLUSION

In this paper, we have shown how both structure and markings can be used as stimuli to regulate the autonomous construction of a staircase. Our approach draws inspiration from the work by Therault and Bonabeau [3] and its significance is reinforced by observational research into the construction behavior of the termite Procornitermes australis [4]. We have implemented these concepts on our real robotic system called SROCS. The concept of SROCS was introduced in our previous paper [5] and consists of a significantly upgraded mobile robot platform and the stigmergic building blocks, which we have discussed in this paper. Our research aims to coordinate multi-robot construction activities using mechanisms inspired by the collective behavior of social insects. To this end, we are working on improving the reliability of our hardware and starting to work with multiple robots. In future work, we aim to generalize our approach so that our system will be able to apply construction rules based on the recognition of substructures in the immediate environment. This generalization will enable us to provide a set of rules to a group of robots that will then perform the specified construction activity.

ACKNOWLEDGMENTS

Marco Dorigo acknowledges support from the Belgian F.R.S.-FNRS, of which he is a Research Director. Navneet Bhalla was partially supported by a postdoctoral fellowship from the Natural Sciences and Engineering Research Council (NSERC) of Canada. We wish to thank Haitham El-faham for help with the design and realization of the first prototypes of the manipulator module, and Anthony Antoun for help with the first prototypes of the stigmergic building block design. Furthermore, we would like to thank the System and Circuit Technology research group at the University of Paderborn for their support with the BeBot platform.

REFERENCES


Fig. 9. Construction of a staircase