2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Daejeon Convention Center
October 9-14, 2016, Daejeon, Korea

Kilogrid: a Modular Virtualization Environment for the Kilobot Robot

Anthony Antoun[1], Gabriele Valentini[1], Etienne Hocquard[2], Bernát Wiandt[3], Vito Trianni[4] and Marco Dorigo[1]

Abstract— **We introduce the Kilogrid, a modular and scalable virtualization environment aimed at swarm robotics research with the Kilobot robot. The main purpose of the Kilogrid is to complement the Kilobots by overcoming some of their limitations (i.e., limited sensors and actuators), making it easier to experiment and to collect data with large groups of robots. The Kilogrid allows researchers to study scenarios featuring a level of complexity that cannot be reached using the Kilobots alone. The Kilogrid is composed of several *modules*, where each module contains four *cells* of 50×50 mm^2. The cells allow for bi-directional communication with the Kilobots. Our first version of a Kilogrid is composed of 64 cells and covers a total area of 400×400 mm^2. We demonstrate the features of the Kilogrid with two case studies in which: *(i)* we extend the sensory system of the Kilobots, *(ii)* we allow the Kilobots to modify the environment, and *(iii)* we collect data (e.g., position, state) from the Kilobots while the experiment is running.**

I. INTRODUCTION

In this paper, we introduce the Kilogrid, a modular and scalable virtualization environment designed for swarm robotics research. The Kilogrid provides a reconfigurable environment for the study of collective behaviors using the Kilobot robot. Our system makes use of the infrared (IR) communication capabilities of the Kilobot to create a virtual environment for sensing, actuation, and data collection. Existing virtualization systems are based on numerous technologies and the specific choice of the technology mostly depends on the sensors of the robot chosen for the experiments (the Kilobot in our case). Similarly to the Kilogrid, some of these systems make use of infrared light communication (e.g., localization [11], self-organized collective decision making [14], [15], task abstraction [3]) while other systems employ RFID tags (e.g., navigation and path planning [6], [7], [16]), ultrasound signals (e.g., communication [17]), or projected light (e.g., pheromone response [1], [5], pattern formation [8]).

The Kilobot [9] is a 33 mm diameter mobile robotic platform designed for research in swarm robotics [2], [4], [12]. Two vibrating motors allow the Kilobot to move on a flat, two-dimensional surface with a maximum speed of 10 mm/s and to rotate on the spot with a speed of 45°/s. The Kilobot is equipped with a light sensor that allows it to measure the intensity of the ambient light, and with an infrared transceiver that allows it to communicate with nearby robots within a range of approximately 100 mm [9]. Once a Kilobot receives an IR message, it can also calculate its distance from the sender.

To prepare experiments, a user can operate the Kilobots using an Overhead Controller (OHC), for example, to start and stop an experiment, program the robots or check their battery voltage. The OHC can also be used to calibrate the motors of the Kilobots and to assign them an ID.

The simplicity, low cost and small size of the Kilobot enable researchers in swarm robotics to use a high number of robots in a limited space. For example, Rubenstein et al. [11] presented a method that allows a swarm of 1024 Kilobots to be programmed in order to aggregate into given shapes. This experiment is the largest swarm experiment ever run so far in terms of number of robots.

However, the simplicity of the Kilobot hardware is often a limiting factor for experiments in swarm robotics. Indeed, the Kilobot has only one sensor to perceive its environment (i.e., the ambient light sensor). Although it has been shown that a swarm of Kilobots can collectively transport objects [10], the Kilobots cannot perceive objects, including the arena borders, and tend to get stuck against them due to their limited thrust. As a consequence, Kilobots might form clusters at the arena borders [15], which alters the spatial distribution of the swarm and might disrupt the swarm performance. This deeply affects both the individual and the swarm dynamics [13], [14]. Additionally, without the support of dedicated tracking infrastructure, data collection can only be performed by logging information within the Kilobot memory and by downloading this information offline after the execution of an experiment using a wired connection to be plugged in each robot. This data collection method is often tedious and particularly time-consuming.

With the Kilogrid, we aim at overcoming these limitations by introducing a virtual environment for the Kilobot. The Kilogrid is a modular system composed of a grid of *modules*, each containing four *cells*. The cells provide bi-directional communication with the Kilobots. Infrared communication is the main feature of the Kilogrid: it provides the Kilobot with the capability to perceive its environment by means of virtual sensors, modify it using virtual actuators, and transmit internal information to the experimenter for data collection. For example, in our experiments, the Kilobots are set to broadcast their ID using their IR transceiver. These IR messages are collected by the Kilogrid and forwarded to the

[1]Anthony Antoun, Gabriele Valentini and Marco Dorigo are with IRIDIA, Université Libre de Bruxelles, Brussels, Belgium. aantoun,gvalenti,mdorigo@ulb.ac.be

[2]Etienne Hocquard is with IRT Jules Verne, Nantes, France, and contributed to this work during a visiting period at IRIDIA. hocquard.etienne@gmail.com

[3]Bernát Wiandt is visiting IRIDIA, and is affiliated with the Department of Networked Systems and Services, Budapest University of Technology and Economics, Budapest, Hungary. bwiandt@hit.bme.hu

[4]Vito Trianni is with the Institute of Cognitive Sciences and Technologies, Consiglio Nazionale delle Ricerche, Rome, Italy. vito.trianni@istc.cnr.it

978-1-5090-3761-2/16/$31.00 ©2016 IEEE

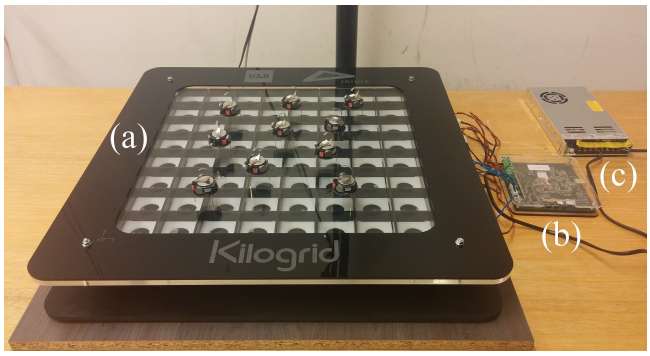

Fig. 1: Picture of the Kilogrid system: (a) Kilogrid of 8×8 cells (i.e., 4×4 modules of 2×2 cells each) with 10 Kilobots on its Plexiglas surface; (b) the dispatcher that interfaces the Kilogrid with a PC; and (c) the power supply.

PC in order to track the position of the robots. Additionally, the Kilogrid provides visual feedback to the experimenter by displaying the state of each cell using colored LEDs. By design, the user can use an arbitrary number of modules in the Kilogrid to cover an area of up to 4 m^2.

We demonstrate the capabilities of the Kilogrid using a system composed of 16 modules covering an area of 400×400 mm^2. We consider two different case studies to show the capabilities of the Kilogrid. In the first case study, we use the Kilogrid to augment the sensory system of the Kilobot. In the second case study, we use the Kilogrid to virtualize actuators that are not otherwise present on the Kilobot. In both case studies, we use the tracking capabilities of the Kilogrid to collect data.

The paper is organized as follows. In Section II, we provide an overview of the Kilogrid and describe its hardware architecture. In Section III, we demonstrate the functionalities of the Kilogrid by performing two series of robot experiments. In Section IV, we summarize the characteristics of our system and discuss future work.

II. Kilogrid

A. Overview

The Kilogrid consists of a grid of modules and of the *dispatcher* that interfaces the modules with a computer (see Figure 1). Once connected together, the Kilogrid modules are placed under a (single) 8 mm thick transparent Plexiglas surface on which the Kilobots move. The Plexiglas surface is the same that was extensively used in previous experiments without the Kilogrid [14], [15]. The modules can be programmed by the user to define complex functionalities in the same manner as for the Kilobots. Each module is divided into four cells, where each cell contains one infrared transceiver and two RGB LEDs driven in parallel. A cell can send and receive infrared messages to and from the Kilobots through the Plexiglas panel. During an experiment, the user can visualize the state of each cell using its RGB LEDs.

The dispatcher incorporates the same functionalities as the original OHC of the Kilobot environment: to program the Kilobots, send them commands such as RESET, RUN, SLEEP, VOLTAGE and PAUSE, calibrate the Kilobots motors and provide them with an ID. In the same manner as with the OHC, one can use the Kilogrid to program the Kilobots through infrared communication, calibrate the Kilobots and send Kilobots commands defined by the kilolib library.[1] In the Kilogrid, the dispatcher is used as an interface between the grid of modules and the computer and allows for bi-directional communication between them. This bi-directional communication is implemented using a Controller Area Network (CAN). The programming environment of the Kilogrid has been developed as an extension of the kilolib library.

B. The Kilogrid module

Figure 2a provides an exploded view of the Kilogrid module. A module covers an area of exactly 100×100 mm^2. The printed circuit board (PCB) of the module lays on four 3D-printed supports and is surrounded by IR barriers that divide the module into 4 equal cells of size 50×50 mm^2. A translucent white Plexiglas panel is placed on top of the cells to diffuse the light emitted by its LEDs. The light diffuser contains four circular holes that allow the propagation of infrared signals between the cell and the robots on its top. Once connected together, the Kilogrid modules are placed under a single Plexiglas panel.

The electronic architecture of the module is illustrated in Figure 2b. The module features an ATmega328P microcontroller unit (MCU) for the execution of user-defined programs. A CAN interface handles all communications with the dispatcher. Our network stack is built on top of the standard CAN protocol in such a way that the modules can communicate with both the dispatcher and with each other. Therefore, the modules are independent and communicating entities with a fixed position on the grid. The CAN interface used in the modules supports up to 112 CAN nodes on the same physical bus. In Section II-C, we explain how we designed the dispatcher to extend the maximum number of modules that a Kilogrid can contain.

As stated above, the Kilogrid module is physically divided into four cells. Each cell contains one infrared transmitter and one infrared receiver, pointing upwards towards the Plexiglas panel. The four infrared transmitters and receivers are multiplexed and independently driven by the MCU. The received infrared signal is processed (i.e., amplified and filtered) in the exact same way as in the Kilobot which permits to measure the distance between a transmitting Kilobot and the cell receiving its message. IR barriers have been added between adjacent cells and modules to prevent cross-talk between cells. The IR barriers are made of laser-cut, 3 mm thick black Plexiglas. The IR barriers between modules are held together by the 3D-printed supports.

Each cell has two RGB LEDs. These two LEDs show the same colors and give the ability to the user to associate an RGB color to each cell. The LEDs have a resolution of 8 bits for each RGB component, which allows to control the color and the intensity of the light.

[1] https://github.com/acornejo/kilolib.

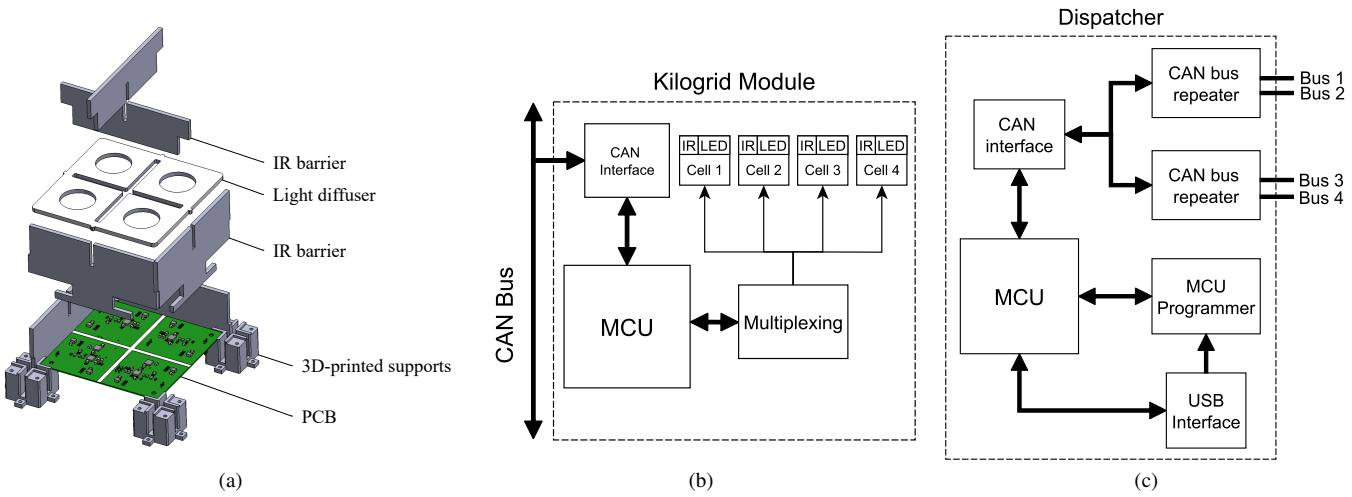

Fig. 2: Illustration of the Kilogrid hardware. (a) 3D exploded view of the Kilogrid module detailing the position of IR barriers, light diffuser, 3D-printed supports, and the PCB. (b) Functional diagram of the cell showing the CAN interface, MCU, multiplexing logic, and the four cells with their IR transceiver and RGB LEDs. (c) Functional diagram of the dispatcher showing the CAN interface with its repeaters, MCU, MCU programmer, and USB interface.

C. The dispatcher

Figure 2c shows the functional diagram of the dispatcher. The USB connection allows to control the Kilogrid with a computer and to program the MCU of the dispatcher via the MCU programmer. The dispatcher is equipped with an ATmega328P microcontroller, similarly to both the module and the Kilobot. Its CAN interface is provided with two CAN repeaters that divide the CAN network into four buses that are physically separated but logically connected. By doing so, we extend the total number of cells that can be connected to the dispatcher to a maximum of $4 \times 112 = 448$. This extension provides us with a theoretical maximum coverage area of 4.48 m^2. Note that this number could be further increased with additional CAN repeaters at the expense of a reduction of the CAN communication frequency.

D. Extension to the KiloGUI application

We extended the existing KiloGUI[2] software so that the Kilogrid can be used to program both modules and Kilobots remotely and to collect data from them, while keeping compatibility with the original OHC. In the current version, KiloGUI records data sent by the modules to the dispatcher (e.g., the position and internal state of the robots) into a data file which is available to the user for offline parsing.

III. Case studies

We demonstrate the capabilities of the Kilogrid using a system composed of 64 cells which covers an area of 400×400 mm^2 (see Figure 1). In all our experiments, we use a swarm of 10 Kilobots. We illustrate the capabilities of the Kilogrid with two case studies. The first case study shows how the Kilogrid can be used to virtualize sensors, while the second case study illustrates how it can be used to virtualize actuators. In both case studies, we use the Kilogrid to collect the data necessary for our following analysis. Video recordings illustrating the results of the robot experiments are available in the supplementary material.[3]

A. Case study 1: Virtual sensors

In this case study, we show how the Kilogrid can provide virtual sensors to the Kilobots so that the experimenter can extend the sensing capabilities of the Kilobot and, by doing so, extend the set of possible collective behaviors that can be implemented and studied. To this end, we present a simple obstacle avoidance experiment. In the light of the limitations mentioned in Section I, we show that *(i)* the sensory capabilities of the Kilobots can be extended through the use of the Kilogrid and *(ii)* the use of the Kilogrid improves the quality of the experiments with Kilobots.

Let us recall that the Kilobots are unable to perceive the presence of passive obstacles. In this case study, we use the Kilogrid to overcome this issue by implementing a beacon system that informs the robots of the presence of the physical walls of the arena (i.e., passive obstacles). To do so, we use the cells at the borders of the Kilogrid as beacons, delimiting the *working area* of the robots. The working area is thus the set of 6×6 cells where the robots can move freely. Note that beacons can be exploited to indicate the presence of any kind of obstacle with a custom shape, and their position is not restricted to signal the borders of the arena.

In our experiments, the robots move following a simple random walk behavior. We compare the distribution of the robots on the arena with and without the beacons to quantify the influence of the beacons on the robots movement.

[2]Kilobotics documentation: http://www.kilobotics.com/documentation.

[3]Available online at http://iridia.ulb.ac.be/supp/IridiaSupp2016-005/.

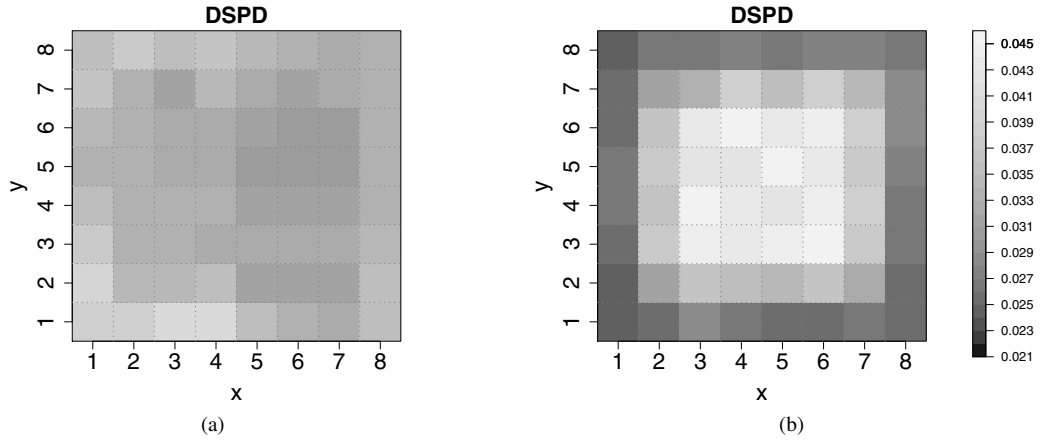

Fig. 3: Illustration of the discrete spatial probability distribution of the robots (p_{ij}) (a) without wall beacons and (b) with wall beacons at the boundary cells. The x and y axes indicate the coordinates of each cell of the Kilogrid. Lighter gray tones correspond to higher probabilities to find robots in a certain position. Data has been obtained via the tracking function of the Kilogrid over 10 runs for each series of robot experiments.

1) Description of the controller: The random walk behavior is implemented as follows. The robot continuously alternates between a FORWARD phase and a ROTATION phase. The duration of the FORWARD phase is normally distributed, with a mean of 3 s and a standard deviation of 1.5 s. The duration of the ROTATION phase is also normally distributed, with a mean of 3 s and a standard deviation of 1.5 s. The direction of rotation is chosen randomly, with equal probability (clockwise or counterclockwise).

When perceiving a wall beacon, the robot stops performing the random walk and enters the wall avoidance procedure. Initially, the robot chooses a random direction of rotation and starts rotating on the spot. This initial choice is made at random because the robot cannot perceive the bearing of the wall beacon. While rotating, the robot receives messages from the wall beacons and uses the strength of the signal carrying each message to estimate its distance from the perceived wall beacon. In order to reduce the uncertainty on the measured distance, the robot aggregates the distances estimated from four consecutive messages by computing their mean value X and uses this information to determine its motion strategy.

Once a robot has computed two consecutive means, X_{old} and X_{new}, its control strategy is defined as follows. When the estimated distance from the border increases (i.e., $X_{new} > X_{old}$), the robot stops rotating and moves straight for a period of time of 2 s to escape the border of the arena. If, during this period of time, the robot does not perceive wall beacons, it has successfully avoided the border and resumes the random walk. Differently, when the estimated distance from the border does not increase (i.e., $X_{new} \leqslant X_{old}$), the robot keeps rotating in the same direction. This direction of motion is kept for at most 7 s after which the robot starts rotating in the opposite direction. This mechanism allows the robot to escape the border of the arena in the case in which it is stuck against it and cannot rotate in the original direction.

During the entire duration of the experiment, each robot broadcasts its ID every 2 s. These messages are perceived by the cells and forwarded to the KiloGUI application by the corresponding modules in order to track the position of the robots in the arena. Due to the fact that IDs are broadcast, the same message can be received by multiple adjacent cells. In this experiment, every time a message from a robot is received by a certain cell, we consider the robot to be positioned on top of that cell.

2) Results and discussion: We performed two separate series of experiments where wall beacons are either enabled or disabled. For each series, we performed 10 independent runs with a duration of 60 minutes each. We track the position of each robot over time and use this information to measure the cumulative residence time T_{ij} spent by all robots over the cell with coordinates (i, j), $i, j \in \{1, \ldots, 8\}$. For each run $r \in \{1, \ldots, 10\}$, we collect all values of $T_{ij}^{(r)}$ for all cells. The discrete spatial probability distribution (DSPD) of the swarm is then defined as

$$p_{ij} = \frac{1}{\tau} \sum_{r=1}^{10} T_{ij}^{(r)},$$

where $\tau = 10 \times 10 \times 60$ minutes is the overall experimental time experienced by the 10 robots in the swarm over 10 runs. Additionally, in order to quantify the influence of the wall beacons on the spatial distribution of the robots, we define the probability P_{work} to find the robots inside the working area as $P_{work} = \sum_{i=2}^{7} \sum_{j=2}^{7} p_{ij}$.

Figure 3 shows the spatial probability distribution of the swarm when wall beacons are not in place and when the wall beacons are present. When wall beacons are disabled, the robots tend to be blocked at the borders of the arena and to form clusters for a certain amount of time (see Figure 3a). This result is confirmed by a relatively low probability of

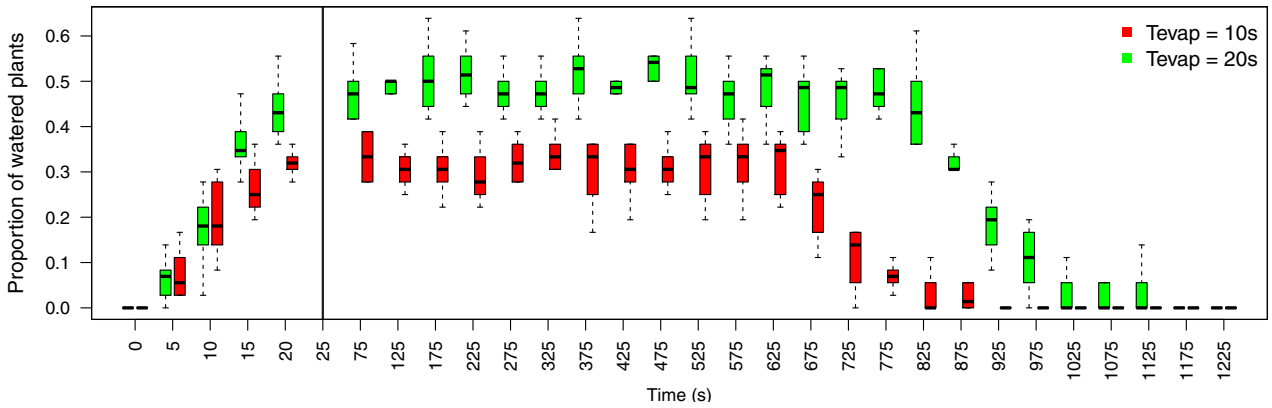

Fig. 4: Illustration of the second case study implementing virtual actuators. We show the evolution of the proportion of watered plants as boxplots over 10 runs for each value of the evaporation time. The red boxplots correspond to an evaporation time $T_{evap} = 10$ s, and the green boxplots correspond to $T_{evap} = 20$ s. The black vertical line identifies the initial 25 s of the experimental time.

finding robots in the working area (i.e., $P_{work} = 0.5039$). Differently, when wall beacons are enabled, the robots can perceive their proximity to the borders of the arena and execute the wall avoidance procedure. As shown in Figure 3b, the probability that the robots are in the working area is much higher when using the wall beacons (i.e., $P_{work} = 0.8157$). The difference in the resulting probability distribution of the swarm in space indicates that the wall beacons prevent the formation of robot clusters at the arena borders.

B. Case study 2: Virtual actuators

The goal of this case study is to show how Kilobots can use virtual actuators to change their environment and react to these changes. Making an analogy with a garden, we consider the robots as "gardeners" and the cells in the working area as plants that need to be watered. Robots search the working area for dry plants to give them water. All plants (i.e., cells) are dry at the beginning of the experiment. Each cell is identified with a unique ID that is used by the robots when sending messages to cells. For the sake of simplicity, and to limit the duration of an experiment, each robot carries 100 units of water, and a dry plant needs 1 unit of water to become wet. Once a plant is watered, it remains wet for a certain period of time called the evaporation time T_{evap}, after which the plant becomes dry again. We consider an experiment completed when all robots have exhausted their water reservoir and all water in the garden is evaporated.

1) Description of the controller: In order to let the robots explore the environment and find dry plants to water, we implemented the same random walk and wall avoidance behaviors previously described in Section III-A using the same parameter values. At the beginning of the experiment, all robots perform a random walk for a time that is uniformly distributed between 0 s and 15 s. This mechanism allows the robots to start watering plants at different times, which prevents collisions (i.e. different robots trying to water the same plant) at the beginning of the experiment.

When a robot detects a dry plant, it moves straight for 1 s to get closer to the plant, then the robot stops on top of the plant for a period of 5 s. During this period, the robot sends messages to the cell representing the dry plant in order to deliver a unit of water. The water capacity of the robot is then decreased by 1 unit and the robot resumes the random walk behavior. The plant that received the unit of water becomes wet for a period of time T_{evap} after which the water evaporates and the plant becomes dry again. When a robot runs out of water, it stops the execution of its controller and notifies the KiloGUI application about this event through the Kilogrid.

In order to measure the number of watered plants over time, we programmed the modules of the Kilogrid as follows. When a cell of a module representing a plant receives a unit of water, the corresponding module of the Kilogrid communicates this information to the KiloGUI application. Similarly, when a unit of water received by a plant evaporates, the corresponding module communicates the occurrence of this event to the KiloGUI application. The KiloGUI application aggregates all information received by the modules of the Kilogrid and outputs the number of watered plants over time.

2) Results and discussion: We performed two separate series of experiments: the first series uses an evaporation time $T_{evap} = 10$ s and the second series uses an evaporation time $T_{evap} = 20$ s. For each series, we performed 10 independent executions of the experiment.

Figure 4 shows, by means of boxplots, the evolution over time of the proportion of plants in the garden that are watered. The proportion of watered plants rapidly increases during the initial 25 s of the experiments as the robots explore the working area and find dry plants to water. After this initial phase, the proportion of watered plants fluctuates around a value that depends on the evaporation time (0.35 for $T_{evap} = 10$ s and 0.5 for $T_{evap} = 20$ s). When the robots begin to exhaust their water reservoir, the proportion of watered plants decreases slowly and converges to zero when all plants

in the garden becomes dry. This happens in approximately 925 s when $T_{evap} = 10$ s and 1175 s when $T_{evap} = 20$ s.

These results show how the Kilobots can act on their environment by means of virtual actuators provided using the Kilogrid. Differently from the previous case study described in Section III-A, we used the Kilogrid to collect data about both the state of the environment (i.e., the Kilogrid) as well as the states of the robots executing the experiment.

IV. Conclusions

In this paper, we presented the Kilogrid, a novel virtualization system for the Kilobot. We showed the benefits of the Kilogrid in two different case studies that illustrate, respectively, how the Kilogrid can be used to extend the sensory system of the Kilobot and how it can be used to create virtual actuators that are not otherwise available on this platform. Throughout these experiments, we also demonstrated how the Kilogrid eases the analysis of robot experiments by providing a means to systematically collect data.

The Kilogrid can be used to provide a robot with the relative position in space of its neighbors. Similarly, a robot may integrate the coordinates of each perceived cell over time to approximate its motion orientation. This approach would provide a coarse-grained virtualization of a bearing sensor. However, the Kilogrid can not provide a robot with its orientation based on the information of a single cell (e.g., if the robot is stuck against a border of the arena). Alternatively, the system could be extended to obtain a more precise virtual bearing sensor with the addition of an overhead camera coupled to the Kilogrid by means of the KiloGUI application. It should be noted that the capabilities of the Kilogrid allow the experimenter to potentially misuse and/or abuse this system by providing robots with unrealistic information (e.g., global knowledge). Nonetheless, the Kilogrid has the potential to mitigate this same issue by offering a standardized experimental environment to the swarm robotics community. When researchers make their controllers publicly available, the Kilogrid can be used to enhance the reproducibility of the results and to facilitate the comparison of alternative control strategies.

We are currently developing a version of the Kilogrid larger than the one presented in this paper that will consist of 800 cells and will cover an area of 1×2 m^2. Using this larger Kilogrid, we aim at performing experiments with a swarm of up to 200 Kilobots to further explore the capabilities of the Kilogrid. We plan on investigating different types of virtual sensors and virtual actuators; for example, sensors and actuators necessary to perform pheromone-based collective behaviors as well as those necessary to follow one or more virtual gradients to navigate the environment. Additional future work includes the use of the Kilogrid to experiment with dynamic environments, a thorough characterization of the IR communication range and bandwidth for both the cells and the Kilobots, and the release under open license of all the files necessary to build the Kilogrid.

Acknowledgments

This work has been supported by the European Research Council through the ERC Advanced Grant "E-SWARM: Engineering Swarm Intelligence Systems" (contract 246939) to Marco Dorigo. Marco Dorigo acknowledges support from the Belgian F.R.S.–FNRS, of which he is a Research Director. The authors would like to thank Yasumasa Tamura of the Hokkaido University for his assistance during experiments.

References

[1] F. Arvin, C. Xiong, and S. Yue. Colias-ϕ: an autonomous micro robot for artificial pheromone communication. *International Journal of Mechanical Engineering and Robotics Research*, 4(4):349–353, 2015.

[2] M. Brambilla, E. Ferrante, M. Birattari, and M. Dorigo. Swarm robotics: A review from the swarm engineering perspective. *Swarm Intelligence*, 7(1):1–41, 2013.

[3] A. Brutschy, L. Garattoni, M. Brambilla, G. Francesca, G. Pini, M. Dorigo, and M. Birattari. The TAM: abstracting complex tasks in swarm robotics research. *Swarm Intelligence*, 9(1):1–22, 2015.

[4] M. Dorigo, M. Birattari, and M. Brambilla. Swarm robotics. *Scholarpedia*, 9(1):1463, 2014.

[5] S. Garnier, M. Combe, C. Jost, and G. Theraulaz. Do ants need to estimate the geometrical properties of trail bifurcations to find an efficient route? A swarm robotics test bed. *PLoS Computational Biology*, 9(3):e1002, 2013.

[6] A. A. Khaliq, M. Di Rocco, and A. Saffiotti. Stigmergic algorithms for multiple minimalistic robots on an RFID floor. *Swarm Intelligence*, 8(3):199–225, 2014.

[7] A. A. Khaliq and A. Saffiotti. Stigmergy at work: Planning and navigation for a service robot on an RFID floor. In *IEEE International Conference on Robotics and Automation – ICRA 2015*, pages 1085–1092. IEEE Press, 2015.

[8] C. Melhuish, J. Welsby, and C. Edwards. Using templates for defensive wall building with autonomous mobile ant-like robots. In *Proceedings of Towards Intelligent Mobile Robots (TIMR99)*, volume 99, 1999.

[9] M. Rubenstein, C. Ahler, N. Hoff, A. Cabrera, and R. Nagpal. Kilobot: A low cost robot with scalable operations designed for collective behaviors. *Robotics and Autonomous Systems*, 62(7):966–975, 2014.

[10] M. Rubenstein, A. Cabrera, J. Werfel, G. Habibi, J. McLurkin, and R. Nagpal. Collective transport of complex objects by simple robots: Theory and experiments. In *Proceedings of the 2013 International Conference on Autonomous Agents and Multi-agent Systems*, AAMAS '13, pages 47–54, Richland, SC, 2013. IFAAMAS.

[11] M. Rubenstein, A. Cornejo, and R. Nagpal. Programmable self-assembly in a thousand-robot swarm. *Science*, 345(6198):795–799, 2014.

[12] E. Şahin. Swarm robotics: From sources of inspiration to domains of application. In E. Şahin and W. Spears, editors, *Swarm Robotics*, volume 3342 of *LNCS*, pages 10–20. Springer, 2005.

[13] V. Trianni, D. De Simone, A. Reina, and A. Baronchelli. Emergence of consensus in a multi-robot network: From abstract models to empirical validation. *IEEE Robotics and Automation Letters*, 1(1):348–353, 2016.

[14] G. Valentini, E. Ferrante, H. Hamann, and M. Dorigo. Collective decision with 100 Kilobots: Speed versus accuracy in binary discrimination problems. *Autonomous Agents and Multi-Agent Systems*, 30(3):553–580, 2016.

[15] G. Valentini, H. Hamann, and M. Dorigo. Efficient decision-making in a self-organizing robot swarm: On the speed versus accuracy trade-off. In *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems*, AAMAS'15, pages 1305–1314, Richland, SC, 2015. IFAAMAS.

[16] J. Werfel. Collective construction with robot swarms. In R. Doursat, H. Sayama, and O. Michel, editors, *Morphogenetic Engineering: Toward Programmable Complex Systems*, pages 115–140. Springer, Berlin, Heidelberg, 2012.

[17] J. Werfel, K. Petersen, and R. Nagpal. Designing collective behavior in a termite-inspired robot construction team. *Science*, 343(6172):754–758, 2014.