Self-reconfigurable Modular e-pucks

Lachlan Murray[1], Jon Timmis[1,2], and Andy Tyrrell[1]

[1] Department of Electronics, University of York, UK
ljm505,jt517,amt@ohm.york.ac.uk
[2] Department of Computer Science, University of York, UK

Abstract. We present the design of a new structural extension for the e-puck mobile robot. The extension may be used to transform what is a swarm robotics platform into a self-reconfigurable modular robotic system. As a proof of concept, we present an algorithm for controlling the collective locomotion of a group of e-pucks that are equipped with the extension. Our approach proves itself to be an effective method of coordinating the movement of a group of physically connected e-pucks. Furthermore, the system shows robustness in its ability to self-reconfigure and self-assemble following a disruption which alters the group's structure.

1 Introduction

Swarm robotics and self-reconfigurable modular robotics are two closely related areas within the larger field of autonomous mobile robotics. Swarm robotics concerns the study of how a collection of relatively simple embodied agents may coordinate their behaviour in a distributed and self-organising manner, whilst relying exclusively on local sensing and communication [17]. Modular robotic systems are also composed of several relatively simple units, however, unlike robotic swarms, the individuals in a modular robotic system may physically connect with one another to form larger robotic structures. An advantage of such systems is that by varying the connectivity of neighbouring units, structures may dynamically transform their morphology to suit their task or environment [19].

The field of swarm robotics is currently far more accessible than that of modular robotics. As is reflected in the availability of both types of hardware. Several swarm robotic platforms are available to buy or have been released as open hardware projects [14, 13, 9, 3, 11, 2]. In contrast, the authors are not aware of any modular robots that are available commercially, and know of only a single open source project [21]. This may, at least partially, be attributed to the differing complexity of the required hardware. Swarm robots are purposefully simple units, whereas modular robots, although simple in comparison to the structures that they may form, require complex electrical and mechanical hardware to facilitate the processes of docking, reconfiguration and inter-robot communication.

To help redress the balance between the two fields and encourage research into modular robotic systems, here we present a low-cost, low-technology extension that may be used to transform an existing swarm robotics platform into a modular robotic system. Our chosen platform is the e-puck robot [14]. As an open hardware project, the e-puck robot is a highly flexible platform. Over recent years a number of extensions have been developed, including an omnidirectional vision turret, a range-bearing board [7], colour LEDs, a ZigBee radio module [1], and even an embedded Linux implementation [12].

In this paper we describe a purely structural extension that allows each equipped e-puck to physically connect with up to four other modules through passive magnetic docking interfaces. The extension may serve as a low-cost and accessible platform for research into the control of 2-dimentional modular robotic systems. As a proof of concept, we present an algorithm for coordinating the motion of a collection of physically connected e-puck robots and observe that our approach is not only amenable to the task, but exhibits robust behaviour in the face of perturbations that disrupt the arrangement of the robots.

The remainder of this paper is structured as follows. In section 2 we provide a short review of existing modular robotics hardware. In section 3 we describe the design of our extension. In section 4 we introduce a proof of concept control algorithm. In section 5 we present results of some preliminary experiments utilising our modular extension. Finally, in section 6 we present our conclusions.

2 Self-reconfigurable Modular Robotics

In 2007, Yim et al. produced a review of the state of the art in modular robotics [19]. The review includes a 'taxonomy of architectures' which classifies platforms as either: *chain*, *lattice*, *mobile*, or if they combine elements of more than one of the previous three, *hybrid*. Platforms may further be classified according to the number of degrees of freedom that the individual units posses, the number of dimensions in which structures can be formed and the method by which they reconfigure themselves, which may be described as *deterministic* or *stochastic*.

In chain-based architectures modules are connected to one another in series but may branch to form tree like structures or fold and reconnect to form loops. The *CKBot* is one such example. Each of the cube shaped modules possess only a single degree of freedom, but as a collective have been shown demonstrate a wide range of movements, notably, including the ability to self-repair following a high impact event that breaks the system into multiple sub-structures [20]. The open source *Molecubes* platform is another example, similar to our goal, the platform was designed to encourage research into modular robotics [21].

Lattice architectures are more restrictive than chain-based systems, with modules only able to occupy discrete positions within a conceptual grid. The *Miche* [4] and subsequent *Smart Pebble* systems [5] are two examples of lattice architectures. Envisaged as a test bed for future systems of *programmable matter*, these small, immobile, cube shaped modules may self-assemble with the help an external stocastic force, for example a vibrating table. Once assembled in a densely packed arrangement, a distributed strategy of self-*disassembly* is used to 'sculpt' the desired object from the robotic substrate.

In a mobile architecture, as well being able to form collective robotic structures, modules are able to move freely around their environment as individuals.

The *s-bot* platform [15], developed as part of the Swarm-bot project, and the robots of the succeeding Swarmanoid project represent the best example of a mobile self-reconfigurable robotic system. The individual robots can physically connect to one another using grippers. Although unable to create structures as complex as those produced by other modular robotic systems, the platforms have been used to developed several distributed control strategies for tackling tasks such as self-assembly [6] and collective recovery [16].

The term "hybrid" is commonly used to describe systems which combine elements from both chain and lattice based architectures. Recently, a new type of hybrid has emerged which also shares some of the properties of mobile architectures. The *Sambot* platform [18], and the robots being developed by the SYMBRION and REPLICATOR projects [10] are two good examples of such mobile-hybrids. Like the Swarm-bot and Swarmanoid projects, the individual robots are independently mobile. However, unlike the s-bot and its derivatives, the modules are also designed to be capable of forming complex 3D structures.

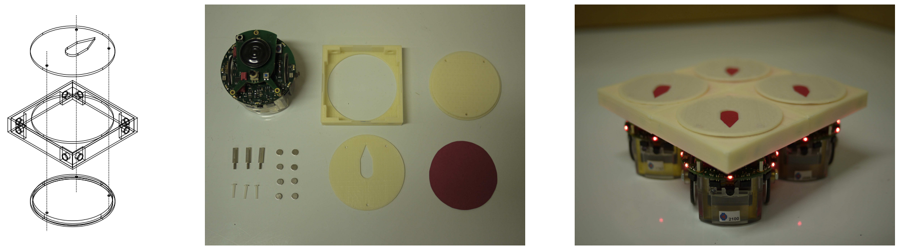
3 Modular e-puck Extension

In this section we present the design of our *modular e-puck extension*. The extension may be used to transform the existing e-puck platform into what can be described as a hybrid *mobile-lattice* modular robotic system. Robots equipped with the extension remain independently mobile, but through passive magnetic docking interfaces may physically connect with other modules within a 2D grid.

As shown in figure 1a, the extension consists of three parts: a circular base plate which sits directly on top of the e-puck, a central frame which rests on top of the base plate, and a second circular plate which sits on top of the frame.

To ensure that there is enough room to clear the selector switch on the default extension board, and to allow access to the reset button, the base plate is positioned on top of three $15mm$ hexagonal spacers. A small overhang on the base plate allows the inner ring of the central frame to rest on the base plate without being permanently attached. This lip allows the frame to rotate unhindered around the central axis of the e-puck. To enable separate modules to connect with one another, two magnets are fitted on each internal edge of the central frame, with opposing poles facing outwards. The strength and positioning of the magnets were chosen such that if connected modules coordinate their motion they will remain attached, but if they do not, they will break apart. Therefore ensuring that the extension provides a suitable platform for investigating both collective behaviour and self-reconfiguration. Screws which pass through the two circular plates secure the extension to the epuck and an arrow shaped window in the top plate allows the current heading of the robot to be easily recognised.

To date, four prototype modules have been produced. The three structural parts of the extension were fabricated using a MakerBot 3D printer. The complete set of parts required to construct a single extension are displayed in figure 1b, we estimate the total cost to be around €5 per unit. Figure 1c shows a potential arrangement of four e-pucks equipped with the fully assembled extension.



(a) (b) (c)

Fig. 1: A schematic of the main structural components of the modular e-puck extension (a) and photographs of unassembled (b) and assembled prototypes (c).

4 Collective Locomotion

In this section we present an algorithm for controlling the collective locomotion of a group of e-pucks that are physically connected using the modular e-puck extension. Through a behaviour-based approach every robot in the group is motivated to move forward, to align with its neighbours and to avoid obstacles. The summation of these three objectives determines the speed of the robot's motors. Regardless of their position within the larger structure, each robot runs the same controller and exchanges information only via local communication.

As a collective, the robots are able to exhibit continuous coordinated motion within an enclosed arena, whilst at the same time demonstrating robustness to perturbations in the overall structure. Following the removal of one or more modules from a group, whether deliberate or accidental, the system is able to self-reconfigure and re-form either the original structure, or an entirely new one. This process of self-assembly is not pre-programmed but emerges due to a combination factors including: the design of the structural extension, the design of the locomotion controller, and the nature of the robot's environment.

The two primary objectives of the controller, to align with neighbouring robots and to avoid obstacles, both make use of the e-puck's infrared (IR) sensors. The arrangement of the eight sensors on a single e-puck is shown in figure 2a. The obstacle avoidance behaviour uses the IR sensors for proximity detection whilst, with the help of the *LibIrcom* library [8], the alignment behaviour uses them for short-range communication.

The alignment behaviour is based upon the same principle of exchanging relative bearings as both the *LibIrcom* library's 'synchronize' example, and the alignment technique described in [7]. We begin this section by describing this method of alignment, from this point on referred to as *static synchronisation*, due to the fact that the robots remain stationary throughout. We identify some problems with this approach when considering non-stationary alignment and whilst introducing a new alignment behaviour propose some solutions. Following which we introduce the obstacle avoidance behaviour and describe how the two parts are combined with a forward bias to produce the desired overall locomotion.

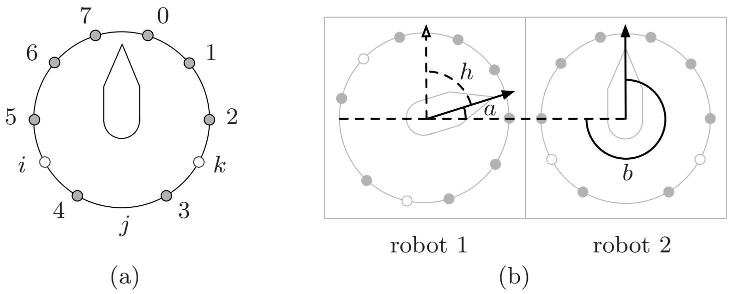


Fig. 2: The positioning of the infrared sensors on board an e-puck robot (a) and the mechanism for exchanging relative bearings between two modules (b).

4.1 Static Synchronisation

The static synchronisation example shows how, by exchanging relative bearings, a group of stationary robots may converge to and maintain a common heading.

Every robot broadcasts its ID and listens for the IDs of others. Based upon the sensor at which a message is received, robots are able to estimate the position of their neighbours as an angle relative to their own heading. For every ID that a robot receives, a message is sent to the corresponding neighbour, notifying it of the angle at which it was detected. As shown in figure 2b, using the angle at which robot 2 was detected (a), and the angle at which robot 2 detected robot 1 (b), robot 1 may calculate the relative heading of robot 2 as $h = a + \pi - b$. The relative heading of each of a robot's neighbours is used to incrementally update the robot's own desired heading, which consequently determines whether a robot should turn left, turn right, or remain stationary at each control cycle.

The approach is effective at synchronising the alignment of stationary robots, but we observe two problems which make it unsuitable for the alignment of mobile robots connected using the modular e-puck extension. Both problems are a consequence of the arrangement of the IR sensors on board the e-puck robot.

Firstly, because the angle between neighbouring sensors ranges from around 30° to 60°, unless two sensors are perfectly aligned, the estimate of angles a and b is often inaccurate. Although the static synchronisation approach incorporates mechanisms for reducing this uncertainty, it is still present. As is evident in the behaviour of the robots, which continuously switch between turning left and right, even once the robots have converged to approximately the same heading.

The second problem is a result of the large gaps between sensors 2, 3, 4 and 5. When two robots are physically connected, the close proximity of the modules and the gaps between the sensors can create blind spots in some orientations (marked i, j and k in figure 2a). As a result of these blind spots, in certain configurations the time taken to converge to a common heading is increased.

The two problems are further highlighted in figure 3. When sending messages via infrared, it is possible to estimate the distance between the sending and receiving sensors by measuring the intensity of the light received. Figure 3a maps the intensity of the infrared signal for messages sent between two robots

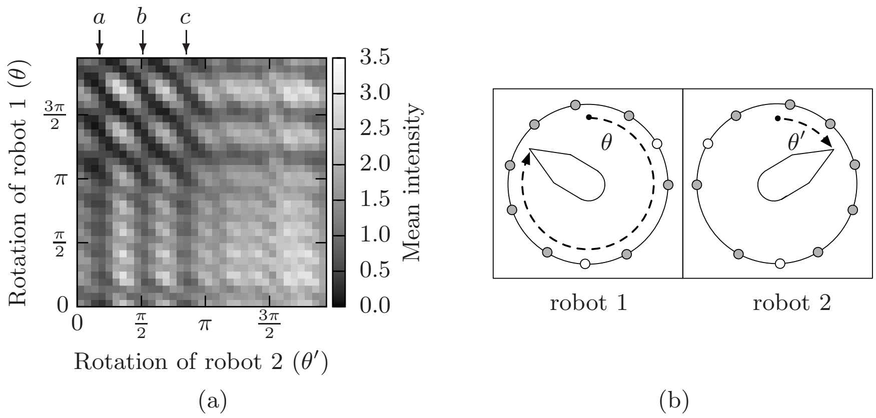


Fig. 3: A map of the intensity of the IR messages sent between two modules at various orientations (a) and a diagram of the setup used to gather the data (b).

arranged at various orientations. The setup used to gather this data is shown in figure 3b where robot 1 is the receiving module and robot 2 is the sender. The intensity of the signal associated with five received messages was recorded at 10° intervals for every 1296 (36×36) possible configurations of the two robots. Where no message was received within a certain time limit an intensity of 0 was assigned. The mean value of the five measurements is plotted. It can be noted from figure 3a that, due to the distribution of the sensors, when the two robots are facing each other (bottom right) the intensity of the received signals is high, but when two robots are facing away from each other (top left) the intensity is often low. A high intensity value indicates that the sending and receiving sensors are closely aligned, so when two robots are facing each other the measurement of angle a and b is likely to be more accurate than when they are facing away.

4.2 Alignment

We now present an alternative approach to alignment which aims to tackle the problems identified in the previous section by making use of the information available in figure 3a. Building upon the static synchronisation approach, robots still broadcast their IDs and track the relative orientation of their neighbours, but as well as making use of the content and direction of the messages they receive, the intensity of the signals also influences their behaviour.

In figure 3a the lines at $x = a$, $x = b$ and $x = c$ correspond respectively to the configurations at which the blind spots i, j and k of robot 2 are directly aligned with robot 1. As shown in figure 3a, the intensity values of the messages received along and adjacent to the lines a, b and c are low. It is possible to make use of this fact to infer when the blind spot of a robot is aligned with its neighbour, and hence to determine the position of the neighbour more accurately. Specifically,

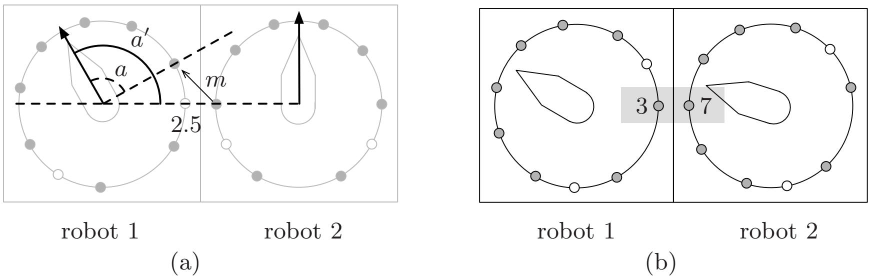


Fig. 4: Diagrams showing the strategy for correcting misalignment using 'virtual' sensors (a) and alignment based upon paired sensors (b).

as shown in figure 4a we may infer that the blind spot k of robot 1 is facing robot 2, when the message m received at sensor 2 reports a low intensity. Whilst it is true that sensor 2 will also report low intensity values when the point between sensors 1 and 2 is aligned with robot 2, because this gap is smaller, these values will never drop as far they do in blind spot k.

A similar inference can be applied to blind spot i and its relation to sensor 5. Notionally then we may define two *virtual* sensors '2.5' and '4.5' which lie between sensors 2-3 and 4-5 respectively. As shown in figure 4a if a message is detected at sensor 2.5, rather than assume it to have originated from a point at an angle a, we may more accurately assume that it originated from an angle a' half way between sensors 2 and 3. Note that it is not possible to define a virtual sensor '3.5' which lies between sensors 3 and 4 because from the perspective of these sensors the blind spots i, j and k are indistinguishable.

It should be noted that, using intensity values alone, it is difficult for a robot to differentiate between the scenarios in which its own blind spot is facing its neighbour, its neighbour's blind spot is facing it, or both blind spots are facing each other. This is not a major concern, however, since in either scenario the reaction is the same, the robots will turn towards each other. Furthermore we may note that messages received from neighbours that are not directly connected, i.e. neighbours positioned at a diagonal, will always have lower intensity values. However, since the *LibIrcom* library preferentially processes high intensity messages, the proportion of messages received from indirect neighbours, and thus the influence they exert, will be lower than that of direct neighbours. In the worst case scenario robots will over eagerly turn towards each other, and as will become apparent in section 5, this is not always a bad thing.

In an attempt to reduce the constant changes in direction witnessed in the static synchronisation example, and to improve the time taken for the robots to converge upon a common heading, we also implement a new method for translating the relative headings of neighbouring modules into motor commands. Rather than incrementally updating an internal desired heading, at each control cycle we calculate the average direction of all the most recently detected headings. This value, h, which belongs to the range $-\pi < h \leq \pi$ is used to determine the speed of the robot's motors. For values of $h < 0$ the robot will turn left and for values of $h > 0$ will turn right, the speed at which the robot turns is proportional to the magnitude of h. For values of $h = 0$ and for control cycles in which no messages are received, the turning speed of the robot's motors is set to zero.

In communicating the relative angle at which a neighbour was detected, the robots transmit the number of the sensor, rather than the angle itself. Furthermore, if $|h| > \frac{\pi}{2}$, indicating that the robot will make a fast turn, to preempt this movement the number of the sensor that is transmitted is incremented or decremented depending upon whether the robot is turning left or right.

Finally, based upon the knowledge that a high intensity signal is indicative of a close alignment between two sensors, we can define certain sensor pairings which, when the intensity of the signal is high, should not influence the movement of the robots. For example, in figure 4b, if robot 1 receives a high intensity message on sensor 3, that was sent from sensor 7 of robot 2, the relative heading of robot 2 will be set to 0. Note that although the alignment between robots 1 and 2 in this scenario is not perfect, it is considered 'good enough' for the task at hand, and preferential to the robots continuously changing direction.

4.3 Obstacle Avoidance

Every sensor which has not recently received a message from another robot, and does not neighbour with a sensor that has recently received a message from another robot, contributes to obstacle avoidance. At each control cycle, the sensors which have detected the presence of an obstacle each create a new desired heading, based upon the position of the sensor. The distance to the detected object is used to assign a weight in the range $(0, 1)$ to each of these new headings, where the closer the obstacle is, the larger the weight. These weighted headings are added to the relative headings of the robots neighbour's, and as before, the average heading h is used to determine the speed of the robots motors. In effect, this is equivalent to assuming that there is a neighbouring robot directly facing every sensor which perceives an obstacle. As well as attempting to align with their neighbours, robots attempt to 'align' with obstacles, with closer obstacles exerting a greater influence over the alignment.

Finally, to ensure that the robots always continue to move forward, we add a small positive bias to the speed of each of the robot's motors.

5 Results

In this section we present the results of a series of experiments conducted using groups of between two and four e-pucks, each equipped with the modular e-puck extension. In the first set of experiments, using a group of stationary robots, we compare the performance of the static synchronisation strategy with our own approach to alignment. After showing our approach to be amenable to the task of stationary alignment, we demonstrate its ability to control a group of mobile robots. At the same time, we observe the robustness of our approach in terms of its ability to recover from perturbations which cause the group to split apart.

5.1 Stationary Alignment

In this set of experiments we compare our approach with the static synchronisation strategy from the *LibIrcom* library. Experiments were conducted using groups of two, three and four stationary robots, arranged as shown in figure 5a.

For each controller and each of the three arrangements, 20 individual runs were conducted. The orientation of the robots was randomised at the start of each run and the absolute heading of each robot was recorded at one second intervals over a period of 100 seconds. Throughout all of the experiments, data was collected using an overhead camera and computer tracking software.

To assess the effectiveness of the approaches in terms of the ability of the robots to converge towards a common heading, we use the same polarisation metric as the authors of [7]. The polarisation $P(G)$ of a group of robots G is defined as the sum of the distance between the heading of every robot and its angular nearest neighbour θ_{ann}. More formally shown by equation 1.

$$P(G) = \sum_{i \in G} \theta_{ann}(i). \tag{1}$$

Figures 5b-d plot the mean polarisation of the two approaches for each of the three module configurations. As is evident by the eventual low polarisation values in all of the figures, in every experimental run the modules were observed to converge to and maintain a common heading. In comparing the two approaches, there is no statistically significant difference between the eventual polarisation of each set of experiments. However, in every configuration, we can observe that convergence is faster for the experiments utilising the new approach to alignment. Furthermore, during the convergence phase (between around 0 and 30 seconds) the variance in the polarisation of the static synchronisation approach greater.

5.2 Collective Locomotion

With the integration of the obstacle avoidance behaviour, we now apply our approach to the task of controlling the collective locomotion of a group of mobile units. The approach was tested for the same configurations used in section 5.1 and, in an enclosed arena ($\sim 0.5 \times 0.7m$), was shown to be capable of effectively coordinating the motion of all three groups. A single run, lasting 30 minutes, was conducted for each configuration. The average position of the robots over the full period is plotted in figure 6. Videos of the experiments are provided online[1].

In all three scenarios the robots were able to successfully navigate the arena without colliding with the arena walls. For the two and four robot configurations all of the modules remained attached to one another throughout. In the three module configuration, for a short period of time one module broke away from the group, only to rejoin soon after. The ability of the module to rejoin the group highlights an important property of our approach, that it is robust to perturbations in the group structure. To further examine this property we conducted

[1] `http://www-users.york.ac.uk/~ljm505/modular_epucks.html`

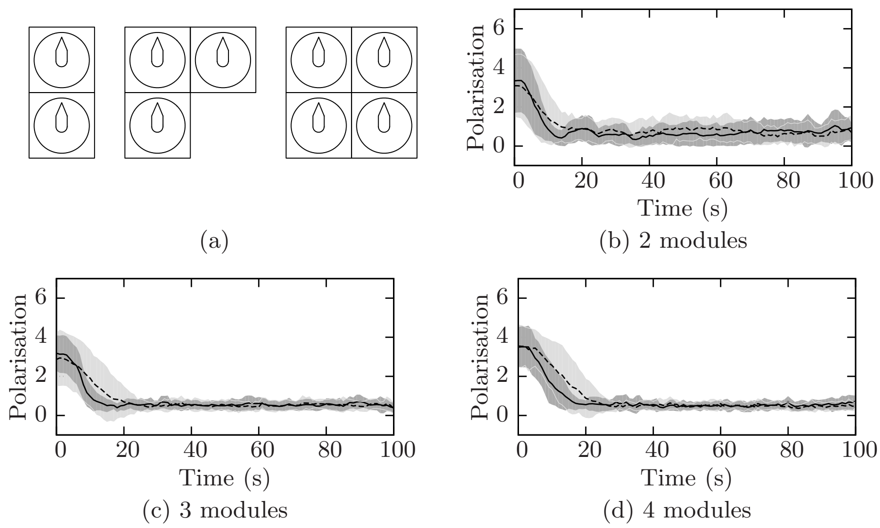


Fig. 5: Figures (b-d) plot the mean polarisation ± one standard deviation, for each of the three configurations in (a). The static synchronisation approach is represented by the dashed line and the lighter grey region, and our new approach is represented by the solid line and the darker region.

another experiment in which three robots were placed in different corners of the arena and left to operate for 10 minutes.

In figure 6d the average pairwise distance between each of the robots is plotted over the 10 minute period. As can be seen in the figure, the robots start far away from one another and gradually converge to a close proximity at around the 5 minute mark. For the remainder of the experiment they remain within close proximity of each other. As shown in figure 6e, what happened in this particular experiment was that at point i two of the modules physically joined together to form a two module structure. Shortly after, at point j the third module joined to complete the three module configuration shown in figure 5a. The robots then remained in this configuration until the end of the run.

It is important to note that this self-assembly behaviour was not pre-programed, it emerges purely due to the interaction of the robots and their environment. Specifically, it can be said to result from a combination of at three factors. Firstly the enclosed arena ensures that robots never stray too far away from one another. Secondly, the alignment behaviour ensures that robots all head in a similar direction. Finally, the design of the e-puck extension ensures that if two robots come into close proximity their magnetic docking interfaces will cause them to 'snap' together. Furthermore, although there is no explicit cohesion behaviour, the implementation of virtual sensors introduced in section 4 may cause robots to move towards each other when they mistakenly believe themselves to be aligned with the blind spot of another robot. These factors combine to produce the semi-stochastic self-assembly behaviour observed in figure 6e.

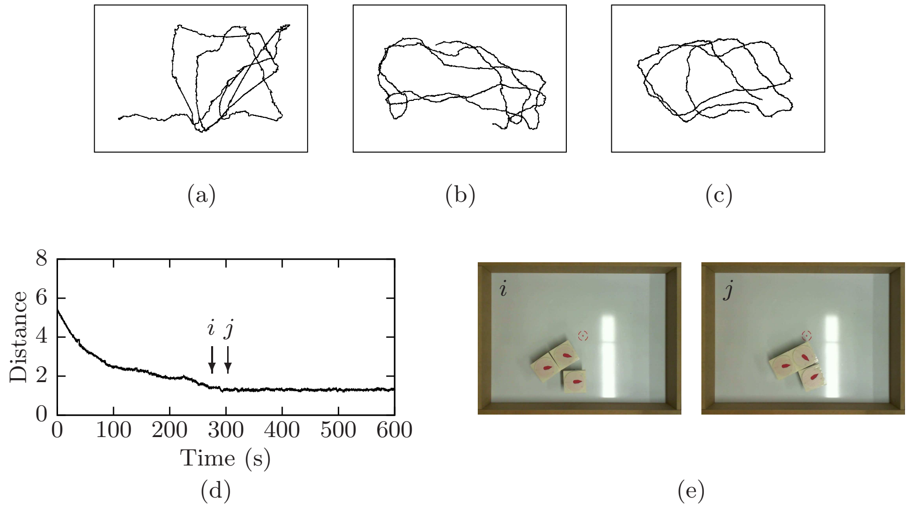


Fig. 6: The average position of groups of two (a), three (b) and four (c) e-pucks equipped with the modular extension, recorded over a 30 minute period, and the pairwise distance between three robots recorded over a 10 minute period (d-e).

6 Conclusions

We have presented the design of a structural extension that may be used to transform the e-puck platform into a *mobile-lattice* modular robotic system. As a proof of concept we described a controller for coordinating the collective locomotion of a group of e-pucks equipped with the extension. The controller was shown to be capable of synchronising the alignment of the group, as well as exhibiting robustness to perturbations which threaten the group's integrity. We conclude that our modular e-puck extension represents a viable low cost platform for research into the control of self-reconfigurable modular robotic systems.

Acknowledgments. The authors would like to thank James Hilder and Martin Trefzer for their invaluable input. The SYMBRION project is funded by the European Commission, within the 7th Framework Programme. FP7-ICT-2007.8.2.

References

1. Cianci, C.M., Raemy, X., Pugh, J., Martinoli, A.: Communication in a swarm of miniature robots: the e-puck as an educational tool for swarm robotics. In: Proceedings of the 2nd international conference on Swarm robotics. pp. 103–115. SAB'06, Springer-Verlag, Berlin, Heidelberg (2007)
2. English, S., Gough, J., Johnson, A., Spanton, R., Sun, J.: Formica (2012), http://formica.srobo.org/
3. GCtronic: Elisa 3 (2012), http://www.gctronic.com/doc/index.php/Elisa_3

4. Gilpin, K., Kotay, K., Rus, D., Vasilescu, I.: Miche: Modular shape formation by self-disassembly. Int. J. Rob. Res. 27, 345–372 (March 2008)
5. Gilpin, K., Rus., D.: Self-disassembling robot pebbles: New results and ideas for self-assembly of 3d structures. In: IEEE International Conference on Robotics and Automation Workshop "Modular Robots: The State of the Art". pp. 94–99 (2010)
6. Groß, R., Bonani, M., Mondada, F., Dorigo, M.: Autonomous self-assembly in swarm-bots. Robotics, IEEE Transactions on 22(6), 1115–1130 (December 2006)
7. Gutierrez, A., Campo, A., Dorigo, M., Donate, J., Monasterio-Huelin, F., Magdalena, L.: Open e-puck range & bearing miniaturized board for local communication in swarm robotics. In: Robotics and Automation, 2009. ICRA '09. IEEE International Conference on. pp. 3111–3116 (May 2009)
8. Gutiérrez, A., Tuci, E., Campo, A.: Evolution of neuro-controllers for robots alignment using local communication. International Journal of Advanced Robotic Systems 6(1), 25–34 (March 2009)
9. K-Team Corporation: K-Team Mobile Robotics (2012), http://www.k-team.com
10. Kernbach, S., Scholz, O., Harada, K., Popesku, S., Liedke, J., Raja, H., Liu, W., Caparrelli, F., Jemai, J., Havlik, J., Meister, E., Levi, P.: Multi-Robot Organisms: State of the Art. In: ICRA10, workshop on "Modular Robots: State of the Art", Anchorage. pp. 1–10 (2010)
11. Kernbach, S.: Jasmine swarm robot platform (2012), http://www.swarmrobot.org
12. Liu, W., Winfield, A.F.: Open-hardware e-puck linux extension board for experimental swarm robotics research. Microprocessors and Microsystems 35(1), 60–67 (2011)
13. Michael Rubenstein, Nicholas Hoff, R.N.: Kilobot: A low cost scalable robot system for collective behaviors. Tech. Rep. TR-06-11, Harvard University (June 2011), ftp://ftp.deas.harvard.edu/techreports/tr-06-11.pdf
14. Mondada, F., Bonani, M., Raemy, X., Pugh, J., Cianci, C., Klaptocz, A., Magnenat, S., christophe Zufferey, J., Floreano, D., Martinoli, A.: The e-puck, a robot designed for education in engineering. In: In Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions. pp. 59–65 (2009)
15. Mondada, F., Pettinaro, G.C., Guignard, A., Kwee, I.W., Floreano, D., Deneubourg, J.L., Nolfi, S., Gambardella, L.M., Dorigo, M.: Swarm-bot: A new distributed robotic concept. Autonomous Robots 17, 193–221 (2004)
16. OGrady, R., Pinciroli, C., Gro, R., Christensen, A., Mondada, F., Bonani, M., Dorigo, M.: Swarm-bots to the rescue. In: Kampis, G., Karsai, I., Szathmry, E. (eds.) Advances in Artificial Life. Darwin Meets von Neumann, Lecture Notes in Computer Science, vol. 5777, pp. 165–172. Springer Berlin / Heidelberg (2011)
17. Şahin, E., Spears, W.: Swarm robotics: SAB 2004 international workshop, Santa Monica, CA, USA, July 17, 2004 : revised selected papers. Lecture notes in computer science, Springer (2005)
18. Wei, H., Cai, Y., Li, H., Li, D., Wang, T.: Sambot: A self-assembly modular robot for swarm robot. In: Robotics and Automation (ICRA), 2010 IEEE International Conference on. pp. 66–71 (2010)
19. Yim, M., Shen, W.M., Salemi, B., Rus, D., Moll, M., Lipson, H., Klavins, E., Chirikjian, G.: Modular self-reconfigurable robot systems [grand challenges of robotics]. Robotics Automation Magazine, IEEE 14(1), 43–52 (March 2007)
20. Yim, M., Shirmohammadi, B., Sastra, J., Park, M., Dugan, M., Taylor, C.: Towards robotic self-reassembly after explosion. In: Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on. pp. 2767–2772 (2007)
21. Zykov, V., Chan, A., Lipson, H.: Molecubes: An open-source modular robotics kit. In: IROS-2007 Self-Reconfigurable Robotics Workshop (2007)