# Evolving Formations of Intentional Robots

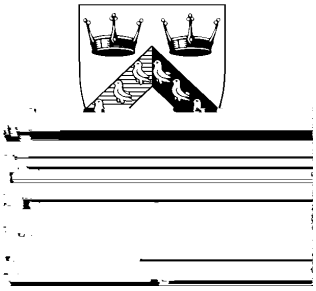
## an Application to Robots Attending to Human Gestures

C.S.R.P. 515

May 2002

ISSN 1350–3162

UNIVERSITY OF



---

Contents

---

# Evo v n For at on ov nt or a Ho o n ous u t o ot st a wor an o A o at on w t a o ots

Matt Quinn, Lincoln Smith, Giles Mayley and Phil Husbands

May 2002

**Abstract**

In recent years a number of researchers have successfully applied artificial evolution approaches to the design of controllers for autonomous robots. To date, how,vol

# 1 Introduction

**In this paper we report on our recent work evolving controllers for robots which are required to work as a team. The word 'team' has been used in a variety of**

heavily on the use of essentially global information, shared by radio communication. For example, in Matarić's implementation of coordinated movement with homogeneous robots, robots made use of a common coordinate system (through radio beacon triangulation) and exchanged positional information via radio communication in order to remain coordinated (Matarić, 1995). Mech-

successful behaviour of one of the evolved teams in some detail, showing that task success is dependent on the robots adopting and maintaining separate roles performing as team, in accordance with definition given at the beginning of this paper.

# 2 The Robots and their Task

## 2.1 The Robots

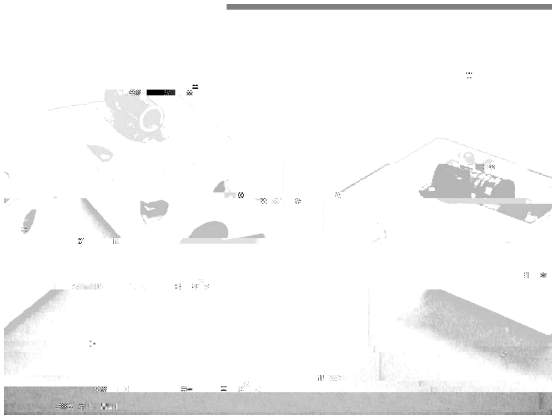


Figure 1: Two of the robots used in the experiment. The cameras shown are not used for the experiments described in this paper.

Two of the robots used in these experiments are shown in figure 1. Each robot's body is 16.75 cm wide by 16.75 cm long by 11 cm high (this excludes the additional height of the camera). Two motor-driven wheels, made of foam-rubber, are arranged one on either side of the robot and provide locomotion through di□erential drive; the robots have a top speed of just over 6cm/s. An unpowered castor wheel, placed rear-centre, ensures stability. In the experiments described in this paper, a robot's *only* source of sensory input comes from its four active infrared sensors, each comprising a paired infrared emitter and receiver. Each robot has two infrared sensors at the front and two at rear, as illustrated in figure 2. Although the robots are also96(pr)-9442(s)-0.333276(,)-353.675(ea)0.42462.a(er)-

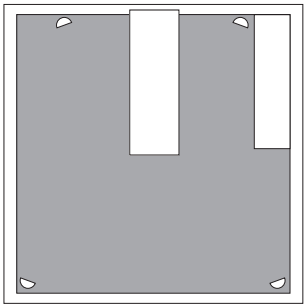



**Figure 2: Plan view of a robot, drawn to scale, showing location of the IR sensors and wheels.**

of the other robots.

## 2.3 The Task

The task with which we present the robots is an extension of that used in previous work which involved two simulated Khepera robots (Quinn, 2001a,b). Adapted for three robots, the task is as follows: Initially, the three robots are placed in an obstacle-free environment in some random configuration, such that each robot is within sensor range of the others. Thereafter, the robots are required to move, as a group, a certain distance away from their initial position. The robots are not required to adopt any particular formation, only to remaining within sensor range of one another, and to avoid collisions. During evolution robots are evaluated on their ability to move the group centroid one metre within the space of three simulated minutes. However, our expectation was that a team capable of this would be able to sustain formation movement of much longer periods. The robots are not required to adopt any particular formation, only to remaining within sensor range of one another, and to avoid collisions. Since the robots start from initial random configurations, we anticipate that successful completion of the task will entail two phases. The first entailing the team organising itself into a formation, and the second entailing the team moving whilst maintaining that formation.

From the characterisation of the robots' sensors in the previous section, it should be clear that these impose significant constraints. They provide very little direct information about a robot's surroundings. Any given set of sensor input can be the result of any one of large number of significantly di□erent circumstances. Furthermore, outside the limited range of their IR sensors, robots have no indication of each other's position. Any robot straying more than two body-lengths from its teammates will cease to have any indication of their location (which can also occur at much closer distances, as can be seen from figure 3). Of course, a robot controller may employ strategies to overcome some of the limitations of it sensors. For example, additional information can be gained by strategies which combine sensing and moving, and the integration of sensor input over time. However, it should be clear that the team's situation contrasts strongly with previous work in which robots utilised shared coordinate systems and global communication. It is worth noting that biological models of 'self-

**That fact that we are attempting design of controllers for a homogeneous sys-**

that all of the beam emitted from the sensor would strike the wall. IR reflected from a wall can be calculated as a function of the distance between sensor and wall, , and angle of the sensor relative to a line orthogonal to the wall, , as illustrated in figure 7 . We took measurements for = 0, 2,...20cm and = -90, -70,...90 degrees (i.e., the full range for which the wall could be sensed). The variation of IR reading with distance for reflected IR is illustrated in figure 5, and an example of how readings vary with the angle is shown in figure 6. Only one set of measurements was taken, that is, we measured one sensor on one robot only.

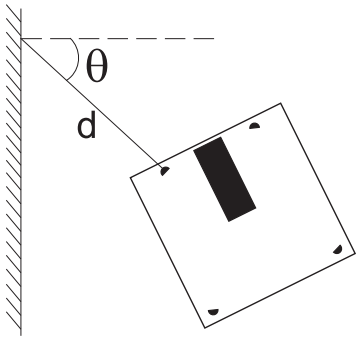



Figure 7: Reflected IR was treated as a function of the distance, , and angle of the sensor relative to a line orthogonal to the wall

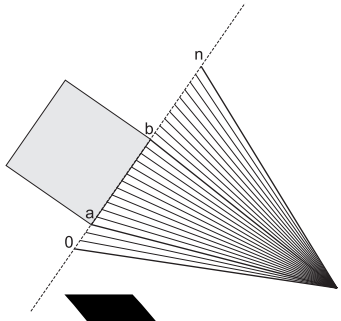
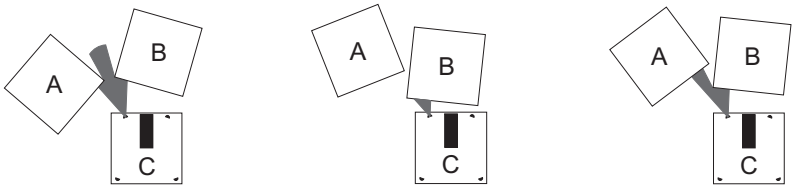



Figure 8: Illus g the 2-d ray-tracing procedure all the rays, 0 through n

60)6

The IR lookup table stores only the sensor values which are generated by the presence of one other robot. Nevertheless, the simulation must be able to deal with the many occasions when two robots are simultaneously within range of a third robot's sensor(s). In such cases, two sets of readings can be taken from the look-up table, one for each of the two robots that are within range of the third robot's sensors. However, the problem arises of how these two sets of values are to be combined to yield a single set of sensory readings. This is a problem because one robot may occlude the other, and the look-up table provides no information as to whether occlusion is taking place. Detecting occlusions and calculating their effect would be a significant additional computational expense in the simulation—it would necessitate ray-tracing across the path of the IR beam. To avoid this, we opted for a minimal and computationally inexpensive solution to the problem. Namely, occlusion is accommodated but not simulated.




i. No occlusion. ii. Full occlusion. iii. Partial occlusion.

Figure 9: When two robots, A and B, are in range of a sensor belonging to a third robot, C, the look-up table gives two independent values for C's sensor; the sensor value due to A's proximity, $\rho_A$, and the value due to B's proximity, $\rho_B$. In an accurate model, occlusion would be taken into account when combining $\rho_A$ and $\rho_B$. Thus, in panel i, where there is no occlusion, C's sensor value should be $\rho_A + \rho_B$. In panel ii, where A is fully occluded by B, C's sensor value should be $\rho_B$. In panel iii, where A is partially occluded by B, C's sensor value should be $\rho_B$, plus some proportion p, (0 , 1), of $\rho_A$.

Consider that two robots, A and B, are both within range of a sensor belonging a third robot, C. Using the look-up table, we can establish C's expected sensor reading due to the presence of A, call this $\rho_A$, and its reading due to B, call this $\rho_B$. We need to establish the the combined, final value, $\rho_{final}$. We proceed by finding the possible range of values within $\rho_{final}$ can lie. The two extremes, non-occlusion and full occlusion, correspond to the extremes of the range of possible values for $\rho_{final}$. If we know that neither robot occluded the other, as in the situation illustrated in figure 9(i), then we could simply to sum the sensor value due to each, i.e. $\rho_{final} = \rho_A + \rho_B$. This is the maximum possible value of $\rho_{final}$

robot. In the case of partial occlusion, illustrated in figure 9(ii), $\gamma_{final}$ will be the value due to the occluding robot plus some proportion ( 1) of the value due to the partly occluded robot. To calculate the exact value due to the partly occluded robot would require ray-tracing across the part of the beam hitting that robot, or some other form of numerical integration. Nonetheless, it is clear that the cases of partial occlusion yield values of $\gamma_{final}$ greater than that of full occlusion, but less than in cases where there is no occlusion. Therefore, given values for $\gamma_A$ and $\gamma_B$ we know range within which $\gamma_{final}$ lies. In the simulation, we set the combined, final value, $\gamma_{final}$ as:

$$\gamma_{final} = \max(\gamma_A \gamma_B) + c_R\left[\gamma_A + \gamma_B - \max(\gamma_A \gamma_B)\right]$$

where $c_R$ is a robot-specific scalar set randomly in the range [0:1] for each robot at the beginning of each trial. Note that to avoid calculating which robot is closest to the sensor we make the approximation that the larger value of $\gamma_A$ and $\gamma_B$ will be the value due to the closest robot[2]. Note that this method will hardly ever produce the correct value for $\gamma_{final}$ and controllers will have to adapt so as to be capable of dealing with any value of $\gamma_{final}$ within range specified above. However, any controller capable of this will be capable of dealing with correct values of $\gamma_{final}$, since these will always lie within that range.

## 3.2 Direct IR

### 3.2.1 Measurement and Extrapolation

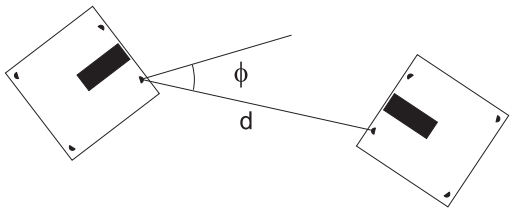
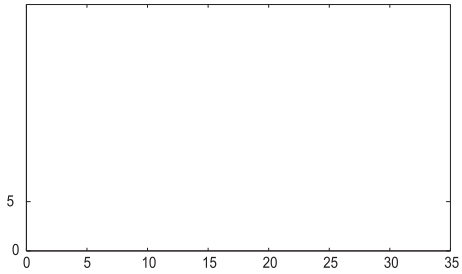



Figure 10: Direct IR was measured as a function of the distance, $d$, and the angle, .

Measuring direct IR was more di cult than measuring reflected IR, since it required lining up two robots' sensors, and eliminating any reflected IR. Readings were taken with the emitting robot's emitter facfunc1l2781D3(em)0.4214-0.240155((i)0.440.42142(he)2


5
0
0
5
10
15
20
25
30
35

**from the closed robot and the sum of the IR reflected from both. However, in the case of direct IR, occlusion can reduce an IR value from the maximum possible value down to zero. Hence, we contrived to model the occlusion of direct IR, albeit as minimally as possible. The procedure we adopted required making some simplifying assumptions, which reduce the accuracy of our model, but made the occlusion-handling process much faster. Our first simplifying assumption is based on the following observation: When one robot receives IR emitted by a second robot, this usually only involves one emitter and one receiver. This is simply due to the positions of the robots' sensors and the properties of the IR beam. In certain positions the beam emitted from one receiver may strike two receivers, or two beams (emitted by the same robot) may strike two separate receivers. However, in such cases, the amount of IR**

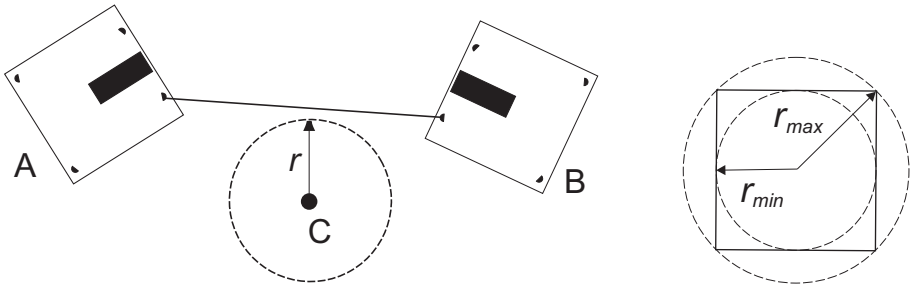



**Figure 13: Checking occlusion by robot C of the beam from A's sensor to B's sensor: The beam is modelled as segment between the two sensors, occlusion occurs if the distance between the centre of C and the beam is smaller than . In the case illustrated, there no occlusion is assumed. is set randomly each trail in the range from $_{min}$, the shortest distance between the centre of the robot and its sides, to $_{max}$, the distance between the robot centre and a corner.**

## 3.3 Combined IR: Final Sensor Values

**At each update of the simulation, each robots' sensors' direct and reflected IR values are established in the manner set out in the sections above. A sensor's direct and reflected values are combined simply by summing them, and capping the result if it exceeded the maximum possible value the could be returned by an IR sensor. The last step in generating a sensor value is the addition of noise. Very little sensor noise is observed on the real robots; sensors return values**

Here is the neuron's threshold. $_t$ is a function of a neuron's weighted, summed input (s), and the value of $_{t-1}$ scaled by a temporal decay constant, such that:

$$_t = \begin{cases} (\ _A)\ _{t-1} + \sum_{n=0}^{N}\ _n\ _n & \text{if } \ _{t-1} = 0 \\ (\ _B)\ _{t-1} + \sum_{n=0}^{N}\ _n\ _n & \text{if } \ _{t-1} = 1 \end{cases}$$

where $_A$ and $_B$ are decay constants, and $_n$ designates the weight of the connection from the $n^{th}$ input ( $_n$) that scales that input. $_A$ and $_B$ are constrained to the range [0:1], the values of weights and thresholds are unconstrained. For certain parameter settings this neuron will behave like a simple spiking neuron, accumulating membrane potential, firing and then discharging (i.e., with

$_A$

entailed that a random Gaussian o□set was applied to each real-valued parameter encoded in the genotype with a small probability, such that the expected number of micro-mutations per genotype was 2.0. The mean of the Gaussian was 0, and its s.d is 0.33 of that parameter's initialisation range. The threshold and weight parameters were unbounded, but the decay constants were restricted to their initialisation range of [0:1]. In cases where the mutated value of a bounded parameter fell outside a bound, its value was set at uniform random to a value between the bound and its pre-mutated value.

Three types of macro-mutation were employed. The first two involve the addition and deletion of genetic material. Ideally we would like to balance the rate at which new genetic material is added to and removed from the population, facilitating steady growth as the added material becomes a functional part of the genotype. We have found that, on average, addition is less disruptive that deletion, so in an attempt to maintain a balance, we have set addition rates lower than deletion rates; individuals subject to addition mutations are more likely to remain in the population than those subject to deletion mutations. The first type of macro-mutation involved the addition or deletion of genes. An addition mutation occurred with a probability of 0.004 per genotype, with the new gene being created and added to the genotype by the same procedure described in section 4.2.1 above, except that the maximum number of connections per connection list was limited to two (in order to minimise disruption). Deletion occurred with probability 0.01, and was applied in one of two ways. With a probability of 0.5, one gene was selected at uniform random and removed from the genotype, otherwise a gene was chosen for removal at biased random, using roulette-wheel selection, with a probability inversely proportional to its age (i.e. the number of generations that it had been in the population)$^5$. The second type of macro-mutation involves the addition and deletion of connections. With probability 0.02, a new connection was created (following the procedure described in section 4.2.1); the connection had an equal probability of being an input or an output, and was added to a randomly chosen gene. With a probability of 0.04, a gene was selected at uniform random, a connection list chosen (with equal probability) and, unless that list was empty, a randomly chosen connection was deleted. The final type of macro-mutation was reconnection. With

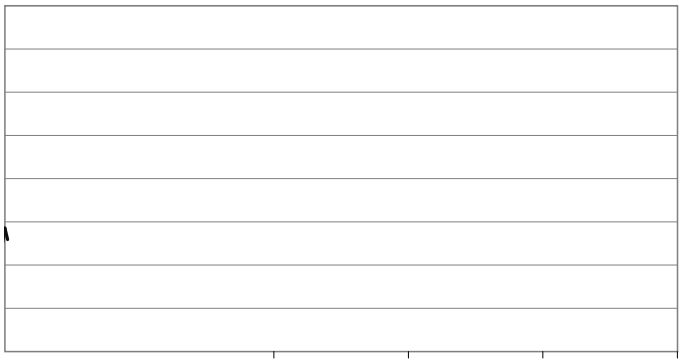
### 4.4.2 Recombination

The recombination (or 'cross-over') operator creates two new 'o□spring' individuals by combining encoded variables from two 'parent' genotypes. Since this is variable-size encoding, the addition and deletion of genetic material means that two genes at the same position on their respective genotypes may not be correlated at the phenotypic level. If crossover were simply based on genotype position, as it is with standard fixed-length encoding schemes, it would often be highly disruptive, with unrelated variables being crossed-over (Jakobi and Quinn, 1998). To avoid this problem, the recombination operator utilises gene's i.d. tags, rather than their position on the genotype, in order to maintain the structural integrity of the resulting genotypes. The i.d. tag is used by the recombination operator to pair genes with a common ancestor, and thereby helps to ensure that genes with similar phenotypic functionality crossed. Crossover takes two parent genotypes, $P_1$ and $P_2$, and generates two o□spring, $O_1$ and $O_2$ in the following manner: Each of o□spring initialised as a copy of a di□erent parents (e.g., $O_1 = P_1$ and $O_2 = P_2$). The o□spring genes are then paired by identity number; any gene that O1 has in common with O2 is crossed (i.e. swapped) with a probability of 0.5. Any remaining un-paired genes are not crossed.

# 5 Evolved Behaviour

To date, we have undertaken a total of ten evolutionary runs. Four of these were terminated at early stage because they seemed unpromising. The remaining six runs produced teams capable of a consistently high standard of success after being left to evolve for between two and five thousand generations. There were significant behavioural di□erences between the successful teams, and we have chosen to focus on a single team rather than attempt to summarise them all. In describing the behaviour of the team, we wish primarily to achieve two objectives. The first is to demonstrate that the robots' behaviour is indeed that of a team, in the sense in which the term was introduced at the beginning of this paper. The second is to illustrate the process by which these roles become allocated in the absence of any intrinsic di□erences between the robots.

It is perhaps a testament to Jakobi's minimal simulation methodology that the controllers we evolved in simulation transferred to the real robots at the first attempt. However, it is also likely that there was an element of luck invone9(rt)03749(f)-313.275lt



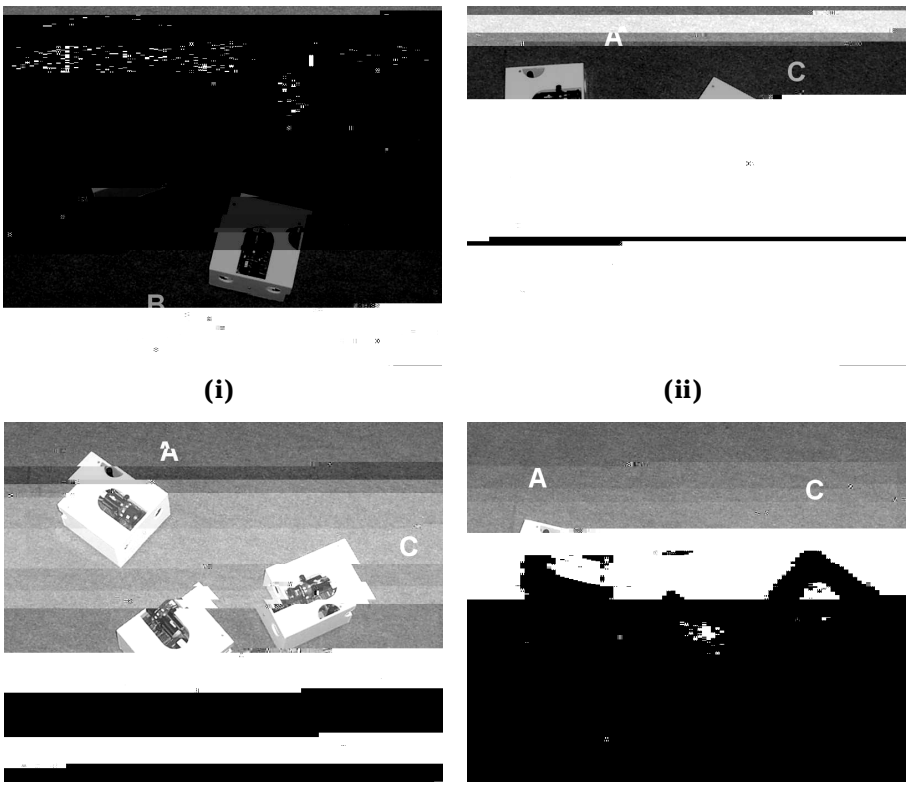



**Figure 20: An example of the team moving into the formation positions. (i) The robot's initial positions. Initially, C is attracted B's rear sensors, causing B to turn tightly, A circles away, clockwise (ii) B and C begin to form a pair as A circles round towards them (iii) A disrupts the pair formation of B and C, subsequently pairing with B. (iv) C becomes attracted to B's rear sensors and begins to move into position. Shortly after this, the team achieve their final formation.**

front robot, formation again move o in the opposite direction, with each robot performing the role appropriate to its position. Thus, the fact that each robot remains in the same role within the formation is solely by virtue of the spatial organisation of the formation, rather than any long-term di erences in internal state[7].

## 5.3 Role Allocation

**How are the roles initially allocated within the team? This is essentially to ask how the robots achieve their formation positions from random initial positions, since as has already been noted, that the maintenance of individual roles is a function of the spatial organisation of the team formation. Any discussion of the initial interactions of the robots will be di□cult without at least some information about how the robots responds to sensory input, so we will start by giving a very simplified explanation. In the absence of any sensory input, the robots move in a small clockwise forwards circle (the motor output is a cyclic**

di□erentiates the team. The excluded robot's role is now determined—it will become the rear robot in the formation. Further di□erentiation occurs when the unpaired robot approaches the back sensors of one of the waiting pair, thereby determining the final two roles.

# 6 Conclusion

The structured cooperation required for the performance of a team task presents interesting problems for a distributed control system. This is particularly true when individuals are homogeneous, and constrained to only make use of limited