The International Journal of Robotics Research
31(6) 753–773

DOI: 10.1177/0278364912442095
ijr.sagepub.com

Image and animation display with multiple mobile robots

Javier Alonso-Mora[1,2], Andreas Breitenmoser[1], Martin Rufli[1], Roland Siegwart[1] and Paul Beardsley[2]

Abstract

In this article we present a novel display that is created using a group of mobile robots. In contrast to traditional displays that are based on a fixed grid of pixels, such as a screen or a projection, this work describes a display in which each pixel is a mobile robot of controllable color. Pixels become mobile entities, and their positioning and motion are used to produce a novel experience. The system input is a single image or an animation created by an artist. The first stage is to generate physical goal configurations and robot colors to optimally represent the input imagery with the available number of robots. The run-time system includes goal assignment, path planning and local reciprocal collision avoidance, to guarantee smooth, fast and oscillation-free motion between images. The algorithms scale to very large robot swarms and extend to a wide range of robot kinematics. Experimental evaluation is done for two different physical swarms of size 14 and 50 differentially driven robots, and for simulations with 1,000 robot pixels.

Keywords

image display, video display, pattern formation, multi-robot system, non-holonomic path planning

1. Introduction

1.1. Motivation

Displays are ubiquitous and range from screens, handheld devices and projection to more experimental technologies such as head-mounted displays (HMDs) and immersive environments. Meanwhile the area of stereoscopic display is generating new ideas such as polarization-based 3D cinema, auto-stereoscopic TV, and lenticular billboards. However, the idea of making a display with physical robots is unexplored. This article describes a display that uses a robot swarm to create representational images and animations, with applications in entertainment. Each robot in the swarm is conceptually one mobile pixel with a RGB LED for controllable color. An example image from a robot display is shown in Figure 1, where 50 robots of custom design display a red fish jumping out of blue water.

A robot display is a new modality with new characteristics. While a traditional display like a screen is a bounded rectangle, and projection requires a surface with suitable physical characteristics, a robot display can extend freely in an environment including onto non-horizontal surfaces (using, for example, magnetic adhesion). This makes it flexible to achieve visibility of the display for a user or a crowd in a variety of configurations. Another distinguishing factor of a robot display is that it not only shows an image, but the styling and motion characteristics of the robots can be used to affect the experience. This is relevant to entertainment applications where robots can, for example, look and move like living creatures in order to interest an audience.

1.2. Contribution

This paper makes three main contributions. First, it introduces the concept of a robotic display to show representational images and animations, and describes a complete system to achieve this. The first stage is goal generation and is done offline: input imagery is used to determine physical configurations and colors of the robots that optimally represent the images. Next, the run-time system involves goal assignment, path planning and local collision avoidance, achieving smooth, fast and oscillation-free motion as the robots transition through images. The algorithms scale to very large robot swarms and extend to a wide range of

[1] Autonomous Systems Lab, ETH Zurich, Switzerland
[2] Disney Research Zurich, Switzerland

Corresponding author: Javier Alonso-Mora, ETH Zurich and Disney Research Zurich, Clausiusstrasse 49, 8092 Zurich, Switzerland.
Email: jalonso@disneyresearch.com

Fig. 1. Fifty robots display a fish jumping out of the water.

robot kinematics. We provide a detailed look at the processing pipeline plus a discussion of scalability and potential for decentralization.

Second, the paper presents two real systems, using 14 and 50 differentially driven robot pixels each. Experimental results are provided for display of static images, display of animations, and handling of perturbations (such as dynamic obstacles or interaction by repositioning robots).

Third, the work is completed by a theoretical analysis on multi-robot pattern formation with discussions on convergence and collision avoidance in real time with large teams of robots.

The remainder of the paper is structured as follows. Section 2 describes related work; Section 3 provides the system overview; Section 4 describes goal generation; Section 5 describes goal assignment, path planning and collision avoidance, with theoretical guarantees; Section 6 describes the extension from static to animated display; Section 7 contains experimental results with physical robots and in simulation; Section 8 concludes the paper and indicates future work.

2. Related work

This paper describes, to the best of the authors' knowledge, one of the first examples of a multi-robot display for showing representational images and animations. The relationship to existing work falls into two categories: conceptually related work on visual effect using a swarm of robots and functionally or algorithmically related work.

2.1. Conceptually related work

Some existing work relates to our goal of visual effect in terms of specific components, although differing in intention. McLurkin and Smith (2004) described distributed algorithms for boundary detection with an illuminated robot swarm moving through an environment. The goal was dispersion and exploration of unknown areas while maintaining local wireless communication. It resulted in equally distributed networks, similar to what is required for our display. Glowbots (Jacobsson et al. 2008) are a swarm of small robots with LEDs that show abstract effects, geometric patterns, and include user interaction. The effect is artistic but there is no mechanism for steering the robots towards a desired configuration to make a specific visual effect. Cianci et al. (2008) demonstrated a group of robotic light sources that adopt different geometric configurations to provide flexible environment lighting. The system uses external infrastructure for localization and we take a similar approach. However, the criterion for deploying the light sources is illumination of the environment and there is no artistic element.

More closely related to our goals, the designer J Tsao envisaged 'Curious Displays' in which hundreds of independent display blocks move around an interior, and aggregate to form integrated displays of various shapes (Tsao 2009). Work on stipple drawing methods addresses image representation by a discrete number of points (Deussen et al. 2000; Secord 2002), but such methods do not treat the problem of image representation with a relatively sparse set of fixed size elements as happens with a multi-robot team. Flyfire (2010) proposed a large swarm of micro helicopters which act as three-dimensional pixels in free space. The work still appears to be in the concept phase, but the animations demonstrate the exciting potential of flying displays. Looking ahead, a robotic display is part of a trend toward pervasive computing and smart objects as miniaturization and autonomy continue to advance.

2.2. Functionally related work

On the functional and algorithmic level, we draw from a large body of work in multi-robot systems that has appeared over the past decade, the most relevant area being pattern formation in large swarms of robots. The key research problems are generation of appropriate target distributions, assignment of individual robots to locations within those distributions, and motion planning of collision-free trajectories. The key challenge is scalability. Excellent reviews on pattern formation methods are found in Bahceci et al. (2003) and Varghese and McKee (2009).

2.2.1. Target distribution generation A common way to form patterns is via the generation of target distributions. Methods which disperse available robots within these distributions are usually iterative, and include the computation of the corresponding control law that drives the robots from their current to their target locations. Potential field analogies have served as inspiration for many algorithms by assigning (virtual) attractive potentials to a prespecified goal distribution and repulsive potentials onto the moving agents. The robots then follow the gradient of the potential until they converge to a local solution. Such a method was explored by Balch and Hybinette (2000) to achieve geometric formations and later by Gayle et al. (2009) to produce impressive simulations, where it was used to drive holonomic robots while avoiding collisions. However, robot kinematics were not respected. Gazi (2005) proposed a method based on artificial potential fields and sliding-mode control in which the kinematics of the robots were considered. Further methods in this category operate directly with filled connected shapes instead of potentials (Ekanayake and Pathirana 2007; Rubenstein and Shen

2010). The latter of these approaches includes autonomous resizing to take into account the number of robots. On the downside, slow convergence is expected due to its distributed nature, with long transitions before the correct size is chosen and the pattern formed. Hsieh et al. (2008) presented a method to represent parametrized curves and (as an extension) arbitrary contours approximated through interpolation. Unfortunately, this method can lead to trap situations in local minima if initial conditions are chosen too far away from equilibrium. As a more general alternative, Voronoi diagrams can be used to generate goal distributions of arbitrary shapes or contours (specified via a density function) as described by Bullo et al. (2009), and this results in optimal coverage. Rounds and Chen (2009) implemented this method to direct a group of robots to zones of high light intensity. However, the iterative control procedure may produce jagged paths and result in slow convergence to local minima.

The localness property of many of the above iterative algorithms can lead to unsatisfactory target distributions, but this can be improved by an appropriate choice of initial conditions (Du et al. 1999). This has two consequences, however. First, the computed control law is no longer valid because the starting conditions do not correspond to the physical placement of the robots. Second, the mapping between robots and their goal location has been severed. To pursue this approach, there is a need for an assignment step between start and goal positions as well as a re-computation of an appropriate control sequence to reach them, see Sections 2.2.2 and 2.2.3.

Many control theoretic decentralized methods achieve formations (control to target distributions) via leader–follower strategies. Important examples include the omnidirectional-camera-based navigation architecture by Das et al. (2002) and the feedback linearization method via non-linear control by Desai et al. (1998). In order to reduce the path planning complexity of the individual robots of a swarm while keeping the ensemble together, a method to steer the global shape of a swarm of robots was developed by Belta and Kumar (2004). This method allows circular and elliptical formations, where the exact position of the robots inside the shape's boundary is of no relevance. An extension of the method to non-holonomic robots was introduced by Michael and Kumar (2008), but in contrast to our work, remains limited to representing elliptic shapes.

Other distributed methods include the following. An earlier distributed behavior-based approach to formation maneuvers was presented by Lawton et al. (2003), where a sequence of maneuvers was used; and a distributed gradual pattern transformation with local information was shown by Ikemoto et al. (2005) for simple patterns and with slow convergence. A behavior-based formation and navigation method in a group of trucks was described by Hsu and Liu (2005). For distributed control methods, keeping the connectivity in the formation is vital to achieving communication between all of the entities. To address this, graph-based methods were studied by Ji and Egerstedt (2007) and Yang et al. (2008).

Finally Takahashi et al. (2009) presented a unique collective approach to pattern formation from computer graphics by defining transitions between agent formations via spectral-based interpolation. The method produces choreographic motions for a group of agents (while collisions are avoided locally using the control functionality of a potential field method). The overall method computes trajectories for all agents individually and seems to require careful tuning.

2.2.2. Assignment Assignment is the mapping of the current robot formation to a target formation. A method to solve this problem optimally was proposed by Kuhn (1955). However, for embodied systems this optimality needs to be traded off with the cost and space requirements of additional computational resources and infrastructure (such as the availability of a common coordinate frame across robots). Thus, suboptimal methods have become popular recently. In this work, we rely on the auction algorithm proposed by Bertsekas (1988) which iteratively converges to the optimal solution. Ravichandran et al. (2007) presented a distributed algorithm for shape transformation based on median consensus. This methods were shown to achieve a performance close to that of the (centrally planned) optimal assignment yet scale well with swarm size.

2.2.3. Motion planning and collision avoidance In case the control law is not computed in conjunction with the target distribution, global motion planning algorithms are well suited to move individual robots between two assigned configurations. Yun et al. (2009) addressed the problem of shape reconfiguration of truss structures. The moderate number of trusses handled, as well as the overall structure of the problem, may have alleviated the difficulty in obtaining solutions. Heuristics become increasingly important for larger sets of robots, and global motion planning algorithms generally do not scale well (LaValle 2006). In the context of displays, however, large non-convex obstacles are typically not present, and the task of motion planning can be substituted with a simpler local collision avoidance method such as the popular velocity obstacles method by Fiorini and Shiller (1998). Much of this method's appeal stems from a recent unique extension to model interactions implicitly (van den Berg et al. 2009). The task of collision avoidance is then distributed in part to each participating agent. This method was further extended to non-holonomic vehicles: differential drive (Alonso-Mora et al. 2010) and car-like (Alonso-Mora et al. 2012), to cope with vehicle constraints that can arise frequently in practice.

3. System overview

This section describes our implementation of a multi-robot display. It is a substantially extended version of the work

presented in Alonso-Mora et al. (2011b). The system consists of a set of robot pixels, an overhead camera, and a central computer that receives images from the camera and wirelessly sends motion commands to the robots. The robots are differentially driven and have RGB LEDs to provide controllable pixel color, and infrared LEDs to facilitate detection and tracking. The goal is optimal placement of the robot pixels to represent images, and smooth, fast-converging motion of the robots. The system displays representational images, as well as abstract images or geometric shapes, and is further extended to display animations. Figure 2 shows the processing pipeline. Goal generation and control are independent components. The algorithms are independent of the kinematics of the robots.

Goal generation is the computation of the robot pixel positions to represent a desired image given a specified number of robots. It is done offline in the current implementation, which is possible when the images are known *a priori*. However, this step could be computed on-line for swarms of the size presented in the experiments of this paper. Goal generation is related to coverage control, and is described in Section 4.

At run-time, a real-time controller drives the robot pixels to the computed goal positions. The multi-layer control scheme is described in Section 5. The controller is iterative and is subdivided into three parts. First, robots are assigned to the goal positions in a unique and optimal way by an auction algorithm. Second, a preferred velocity towards its assigned goal position is computed for each robot independently. Finally, a distributed reciprocal collision avoidance algorithm finds a collision-free velocity for each robot taking into account its kinematics and the current positions and velocities of its neighbors. The new velocities are close to the robots' preferred velocities, and enable a safe motion update of the robotic display. As shown by Latombe (1991) it is intractable to compute the optimal motions of N robots interacting in a common workspace because the search space is exponential in N. Thus, a distributed scheme, such as that we propose, is needed for scalability to large swarms of robots.

4. Goal generation

A set of goal positions and colors for the robots to optimally represent a given image $\mathcal{I}$ is initially computed, based on Voronoi diagrams and methods from locational optimization (Okabe and Suzuki 1997). Centroidal Voronoi tessellations (CVTs) have been rediscovered for robotics by Cortes et al. (2004) by deriving decentralized control laws for robotic environment coverage. In this work, the CVT is applied for iterative optimization of the robots' goal positions.

4.1. Image segmentation

The input image is given as the color map $\mathcal{I} : \mathcal{Q} \rightarrow [0, 255]^3 \subset \mathbb{N}^3$ which assigns a color $\mathcal{I}(q)$ to each position of a normalized square $\mathcal{Q}$, with $q \in \mathcal{Q} = [0, 1] \times [0, 1] \subset \mathbb{R}^2$.

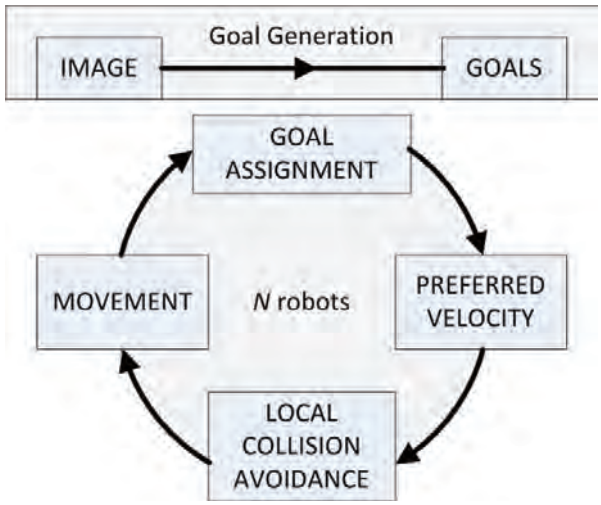

Fig. 2. System overview: goal generation and control loop.

First, a background subtraction is applied to the image assuming only the foreground is to be represented. Second, the foreground is segmented into M connected regions $R_i \subset \mathcal{Q}$, satisfying $R_i \cap R_j = \emptyset$, $\forall i \neq j \in I_R = [1, M] \subset \mathbb{N}$. The region $\mathcal{Q} \setminus \cup_{i \in I_R} R_i$ is considered as empty space, where no goal positions are distributed. This method gives good results for representational images such as those presented in this work. Nevertheless, the segmentation of a real image into an optimal number of representative entities, that allow for an accurate image representation, is a fundamental but still unsolved problem in computer vision (Szeliski 2011).

4.2. Initial samples

Given the total number of robots N, N_i goal positions are to be found for each region R_i, satisfying $N = \sum_{i \in I_R} N_i$. In our system we make the assignment proportional to the area of each region A_i. Here $N_i \propto A_i$ enables a balanced density of goal positions over all regions of interest. Alternatively, they can be defined by an artist.

We sample in the following from a uniform distribution inside each R_i while accepting position values $\mathbf{q}$, $\mathbf{p}$ if $\|\mathbf{q} - \mathbf{p}\| \geq K_s$, $\forall \mathbf{p}, \mathbf{q} \in R_i$, where K_s is a constant, in this work set to $1.2\sqrt{\frac{A_i}{\pi N_i}}$, where A_i is the area of region R_i, $\sqrt{\frac{A_i}{\pi N_i}}$ the radius of a circle of area $\frac{A_i}{N_i}$ and the factor 1.2 is chosen in order to obtain a good distribution with low computational cost (see Figure 3 on the left). The proposed method is similar to generating a Poisson disk distribution inside the regions R_i (see Lagae and Dutré 2006).

The iterative optimization of the goal positions for each region by a Voronoi coverage algorithm converges to configurations of local optima, with asymptotic convergence for any start configuration of generators (Bullo et al. 2009). Hence, a good choice for the initial goal positions not only

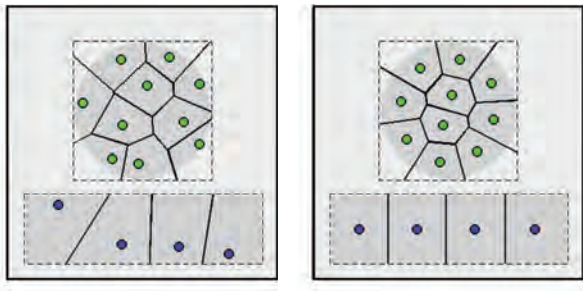

Fig. 3. *Left*: Initial samples for representing a disk and a rectangle with according Voronoi tessellation. *Right*: Final positions of the samples representing the resulting goal positions after convergence, and their Voronoi tessellation. Regions R_i and $\hat{R}_i$ of the image are displayed in gray filling and dashed line, respectively.

results in faster convergence but also in convergence to better distributions (Du et al. 1999).

4.3. Iterative optimization

After initialization of the samples, an iterative optimization based on the Lloyd algorithm (Lloyd 1982) and the CVT computation (Du et al. 1999) is performed independently for each of the M regions. The cost function to be minimized is given by

$$\sum_{j=1}^{N_i} \int_{V_j^i} \|\mathbf{q} - \mathbf{q}_j^i\|^2 \rho_i(\mathbf{q})\, d\mathbf{q}, \tag{1}$$

where the set $\{V_j^i\}_{j\in[1,N_i]}$ is a partition of the convex region $\hat{R}_i$ that entirely encloses R_i, $\mathbf{q}_j^i$ are the samples, which represent the goal positions and act as generators of the partition and $\rho_i(\mathbf{q})$ a mass density function which takes high values in R_i and decreases towards zero outside. In our implementation the set $\mathcal{Q}$ is divided into a grid $A = \bigcup_{r,s} A_{r,s}$, equivalent to the pixels of the original image and the mass density functions are defined as $\rho_i : A \cap \hat{R}_i \longrightarrow \Re_+$, where, with an abuse of notation, $q \in A_{r,s}$ and

$$\rho_i(\mathbf{q}) = \begin{cases} K_m & \text{if } A_{r,s} \cap R_i \neq \emptyset \\ K_d \max_{\hat{\mathbf{q}}\in A_{r\pm\{0,1\},s\pm\{0,1\}}} \rho_i(\hat{\mathbf{q}}) & \text{otherwise.} \end{cases} \tag{2}$$

Here $\rho_i(\hat{\mathbf{q}})$ is the value of the mass density function in a neighboring grid cell of $A_{r,s}$, assuming 8-connectivity of the grid. Uniform distributions of goal positions inside the regions of interest are favored by mass density functions of extremely steep gradient, where $\rho_i(\mathbf{q}) \to \infty$, $\forall \mathbf{q} \in R_i$ and $\rho_i(\mathbf{q}) \to 0$, $\forall \mathbf{q} \in \hat{R}_i \setminus R_i$. Accordingly, the values for our choice of $\rho_i(\mathbf{q})$ are selected as $K_m = 10^{15}$ and $K_d = 0.1$.

For each region R_i, starting from the sampled initial configuration of the goal positions $Q_i = \{\mathbf{q}_j^i, j \in I_i = [1, N_i]\}$, each cell of the Voronoi partition $\mathcal{V}(Q_i) = \{V_1^i, \ldots, V_{N_i}^i\}$ of $\hat{R}_i$ is given by

$$V_j^i = \left\{ \mathbf{q} \in \hat{R}_i \mid \|\mathbf{q} - \mathbf{q}_j^i\| \leq \|\mathbf{q} - \mathbf{q}_k^i\|, \forall j \neq k \in I_{R_i} \right\}. \tag{3}$$

For the mass density function ρ_i, the mass centroids of the resulting Voronoi regions are given by

$$C_j^i = \frac{\int_{V_j^i} \mathbf{q}\rho_i(\mathbf{q})\, d\mathbf{q}}{\int_{V_j^i} \rho_i(\mathbf{q})\, d\mathbf{q}}, \quad \forall V_j^i \in \mathcal{V}(Q_i). \tag{4}$$

Finally, the current position of the goal is updated to the centroid of its region, $\mathbf{q}_j^i = C_j^i$.

After convergence, a final configuration $\mathcal{P}_G = \{\mathbf{q}_j^i, i \in I_R, j \in I_i\}$ with goal positions uniformly distributed in each of the regions of interest is obtained (see Figure 3).

As a simplification, the mass density of Equation (2) can be substituted by a binary distribution taking value 1 in R_i and 0 otherwise. In this case, the CVT iterative optimization can be computed by a k-means algorithm (Inaba et al. 1994) acting on the pixels of R_i (Du et al. 1999). This method presents faster computation and is used in Section 7.5.

4.4. Resizing of generated goal positions

The goal positions were computed for a unit square grid $\mathcal{Q} = [0, 1] \times [0, 1]$ for generators of zero size. In order to account for the robots' finite physical size, resizing is required. The set of goal positions is given by $\mathcal{G} = \{K_r\mathbf{q}, \mathbf{q} \in \mathcal{P}_G\}$, where K_r is the size of the display arena. To guarantee feasibility of the goal positions, the minimum width K_r must satisfy $K_r \geq 2r_A \frac{1}{d_{\min}(\mathbf{p},\mathbf{q})}$, where r_A is the radius of the robot and $d_{\min}(\mathbf{p}, \mathbf{q})$ is the minimum distance between any pair of goal positions. Following a more conservative approach, selecting a value of $2K_r$ furthermore ensures accessibility to all goal positions.

4.5. Complexity and convergence analysis

The centralized implementation of the CVT presents $\mathcal{O}(N_i \log N_i)$ overall time complexity of one step of the computation for N_i particles, as stated in Deussen et al. (2000), where a sweep line algorithm is used to compute the Voronoi diagram. Like in our experiments, it was observed that only 10–20 iterations were needed to obtain a visually appealing distribution with all of the goal positions uniformly distributed on the shape. The CVT can also be computed with a distributed algorithm presenting $\mathcal{O}(N_i)$ time complexity for each robot and step.

Asymptotic convergence is guaranteed to a local minima of the cost function (Bullo et al. 2009), which provides a good measure of the representation of the image. We show from experimental results that fast convergence to a near-minima is achieved due to the proposed initialization of starting positions.

Figure 4 presents the normalized value of the cost for a distribution of 14 samples of goal positions in successive

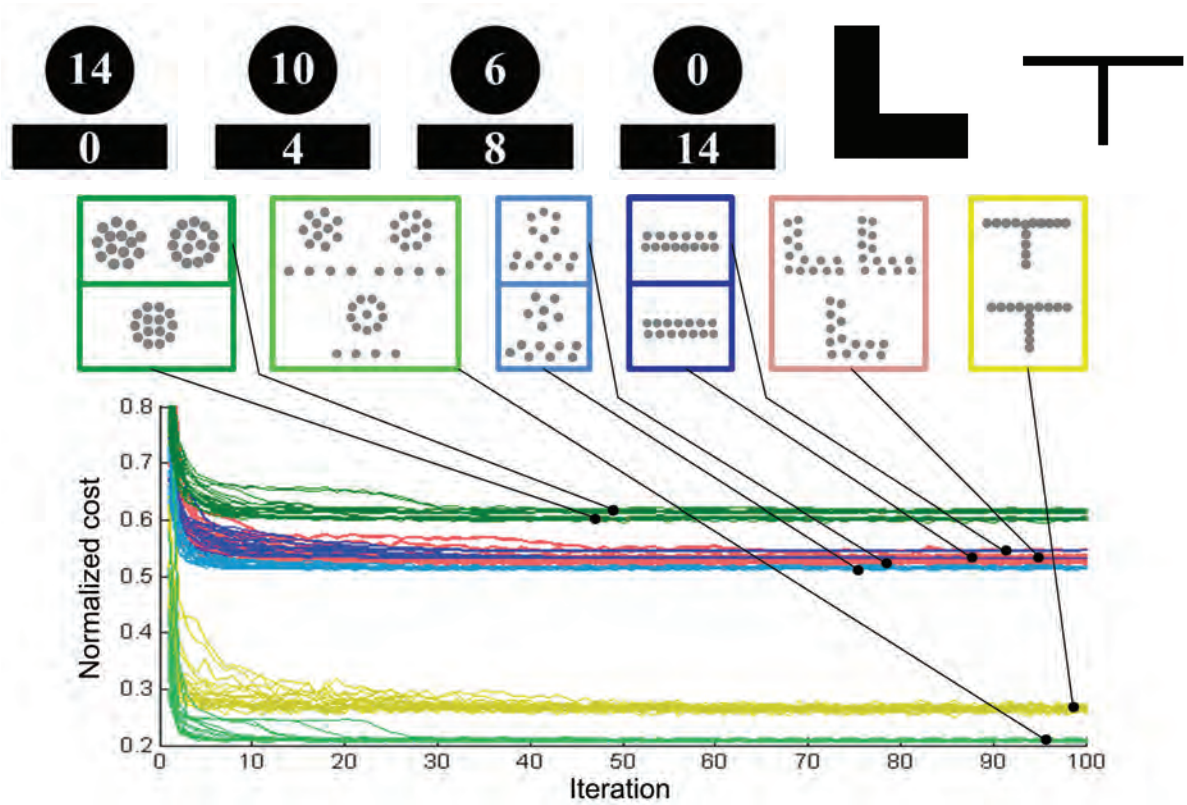

Fig. 4. *Top:* Sequence of input images: disk and line (number of robots specified on top), L-shape and T-shape patterns. *Middle:* Representative sets of goal positions for 14 samples after convergence of the iterative optimization. *Bottom:* The normalized cost of the goal distributions is displayed (in different colors for each image) at each iteration of the optimization. The optimization is performed 25 times for the 6 images starting from randomized initial configurations as described in Section 4.2. For each iteration, the normalized cost is given by the cost of the distribution (Equation 1) divided by the maximum cost of the initial 25 configurations for that image.

steps of the iterative optimization. For 6 different simple images, a set of goal positions is computed 25 times with independent randomized initializations (Section 4.2). For each image, the cost at each iteration is computed from Equation (1) and normalized by dividing it by that of the initialization with the highest cost (out of the 25 runs for the given image).

For all images and initializations a fast decrease in the cost is observed, which leads to final goal positions close to local minima of the cost function within 10–30 iterations. The resulting final goal positions are a good representation of the given patterns. However, for the same image different cost levels are observed after convergence (different local minima). In the top row, the six different images are displayed. Below them, some representative sets of goal positions obtained after convergence are shown. These goal configurations are related to their respective cost plots (lines in Figure 4 show the connections between configurations and cost levels). For the disk, three representative configurations were found, where the lowest represents lowest cost. Similarly, for the third and fourth images, two clearly separated levels of cost resulted, with that of the lowest cost being the most representative. For the second image, the L and the T images, again different representative configurations were identified, although with not as clear separation in the cost level. These observations justify our choice of the cost function from Equation (1) as a measure for good representation of an image.

As shown in the examples, the method performs well for images of both convex and non-convex objects. In this work, the concavities of non-convex regions R_i remain accessible and thus represent weak constraints only. In the context of Voronoi coverage, several extensions to non-convex environments have been studied, where the non-convex regions are due to obstacles and thus impose hard constraints of avoiding them. Pimenta et al. (2008) suggested the change of the distance measure for the geodesic distance, and Breitenmoser et al. (2010) approached the problem by projecting the centroids of the Voronoi cells into accessible space. In our work, thanks to the initialization of the goal positions inside the shapes, good representations of the images are achieved even for non-convex objects and therefore these extensions are not required.

5. Real-time control

In this section the controller for real-time image display is explained in detail. Recall that at this stage a set of goal positions has been obtained (see Section 4). The controller consists of the following steps: goal assignment, computation of a preferred velocity and collision-free inputs. Finally, convergence is proven under certain conditions.

Note that in this section the subindex $i \in I = [1, N]$ represents the robot index, $j \in I$ the goal position index and $k \in \mathbb{N}$ the time-step index of the controller. Let $\mathbf{p}_i^k$ denote the position of robot i at time step k and $\mathcal{P}_k = [\mathbf{p}_1^k, \ldots, \mathbf{p}_N^k]$ the set containing the positions of all robots at time step k.

5.1. Goal assignment

In each iteration each robot i is uniquely assigned to a goal position $\mathbf{g}_j \in \mathcal{G}$, so that a global cost function C (see Equation (7) below) is minimized.

The goal assignment function $\hat{\sigma}_k^*$, which uniquely assigns each robot to a goal position, is defined as

$$\hat{\sigma}_k^* = \underset{\hat{\sigma}_k}{\operatorname{argmin}} \quad C(\hat{\sigma}_k), \tag{5}$$

where

$$\begin{aligned} \hat{\sigma}_k : \quad \mathcal{P}_k &\longrightarrow \quad \mathcal{G} \\ \mathbf{p} &\longmapsto \quad \mathbf{g} \end{aligned} \tag{6}$$

is a bijective map between the current robot positions and the goal positions. Thus, the assignment function $\hat{\sigma}_k$ can also be defined as a permutation σ_k of the elements of I, where $\hat{\sigma}_k(\mathbf{p}_i^k) = \mathbf{g}_{\sigma_k(i)}$. In particular, $\hat{\sigma}_k^*(\mathbf{p}_i^k) = \mathbf{g}_{\sigma_k^*(i)}$.

The cost function is defined as the sum over all robots of the squared distance to their respective goals,

$$C(\hat{\sigma}_k) = \sum_{i \in I} \|\mathbf{g}_{\sigma_k(i)} - \mathbf{p}_i^k\|^2, \tag{7}$$

where $\|\mathbf{x}\|$ is the Euclidean norm of $\mathbf{x}$. The sum of the distances may also be considered as a cost function, but its main disadvantages are an increase in time to convergence within the group of robots, and as further discussed in Section 5.5.4, deadlocks are more likely to appear.

As presented in Alonso-Mora et al. (2011b) an optimal solution of this classical assignment problem is given by the centralized *Kuhn–Munkres assignment algorithm* (Kuhn 1955) which presents at best $\mathcal{O}(N^3)$ cost and is computationally too expensive for large groups of robots. Alternative methods are based on the *auction algorithm* (Bertsekas 1988). These last methods produce suboptimal solutions, as close to the optimum as desired, in significantly lower time. Moreover, they scale very well with the number of robots and can be distributed. In our experiments with large swarms of robots the auction algorithm resulted in drastically reduced computation time compared with the optimal Kuhn–Munkres method. Decentralized versions with both synchronous and asynchronous algorithms were also presented by Bertsekas and Casta–on (1991) and Zavlanos et al. (2008). An alternative distributed method using nearest-neighbor allocation was used by Cianci et al. (2008). The distributed computation is desirable for certain realizations of a multi-robot setup but is not required in our system. In this work a Jacobi-version forward auction with ϵ_G scaling (Bertsekas 1988) is implemented and an overview is given in Section 5.1.1.

For scenarios with strong noise component, the goal assignment scheme can be modified by adding a hysteresis factor. The optimal assignment at iteration k is then compared with that obtained in the previous iteration and only kept if it represents a decrease in cost of at least ΔC, a factor depending on the noise component. This also avoids undesired oscillations due to the suboptimality of the goal assignment.

5.1.1. Auction algorithm The Jacobi forward auction algorithm presented by Bertsekas (1988) proceeds in iterations and generates a sequence of price vectors and assignments. In each iteration the set I_A of unassigned robots is considered and the algorithm proceeds until all robots are assigned. The costs are given by $c_{ij} = \|\mathbf{g}_j - \mathbf{p}_i^k\|^2$ and the prices or bids b_j are initially set to one for all goals. Each iteration consists of two phases.

Bidding phase. For every goal, the list of robots bidding to it is initialized, $B(j) = \emptyset$. Each robot $i \in I_A$ finds the goal with maximal value $j_i = \underset{j \in I}{\operatorname{argmax}}\{-c_{ij} - b_j\}$, is added to the list $B(j_i) \leftarrow i$, and computes a bidding increment $\gamma_i = (-c_{ij_i} - b_{j_i}) - \max_{j \in I, j \neq j_i} \{-c_{ij} - b_j\} + \epsilon_A$.

Assignment phase. Each selected goal j determines its highest bidder $i_j = \max_{i \in B(j)}\{\gamma_i\}$, raises its price $b_j = b_j + \gamma_{i_j}$ and is assigned to robot i_j.

The algorithm is applied several times while decreasing the parameter ϵ_A and keeping the prices of the previous iteration. It is guaranteed that the assignment of each robot differs in less than ϵ_A from the optimal assignment and that the algorithm completes in finite time. For details, refer to Bertsekas (1988).

5.1.2. Multiple goal sets The transformation of an image into a set of goal positions offers the advantage of faster computation and enables the real-time application for large swarms, but it also presents a disadvantage, which is a decrease in flexibility of representing a given pattern.

To overcome this problem, several sets of goal positions can be computed for a given pattern and changing number of robots,

$$\hat{\mathcal{G}} = [\mathcal{G}_1, \ldots, \mathcal{G}_{N_g}], \tag{8}$$

where N_g different sets of goal positions $\mathcal{G}_{s \in [1, N_g]}$ are independently generated with the algorithm of Section 4. Each set $\mathcal{G}_r$ is generated for r robots, and thus contains r goal positions representing the given image. This increases the

flexibility of the system by making it able to adapt to a changing number of robots in real time. Although this is implemented in our system, for clarity of exposition, a single set, $N_g = 1$ for N robots is considered in the remainder of this article.

5.2. Preferred velocity

In each timestep k, each robot i first selects a preferred velocity $\mathbf{v}_{pref_i}^k$ without taking its kinematics or the other robots into account.

The ideal preferred velocity $\mathbf{v}_{pref_i}^k$ is given by a simple proportional controller towards its assigned goal

$$\mathbf{v}_{pref_i}^k = V_p \min\left(1, \frac{\|\mathbf{g}_{\sigma_k^*(i)} - \mathbf{p}_i^k\|}{K_a}\right) \frac{\mathbf{g}_{\sigma_k^*(i)} - \mathbf{p}_i^k}{\|\mathbf{g}_{\sigma_k^*(i)} - \mathbf{p}_i^k\|}, \quad (9)$$

where the constant $V_p > 0$ is the preferred speed of the robot and $K_a > 0$ the distance to the goal from which the preferred velocity is reduced linearly. In order to guarantee convergence without oscillations, K_a must verify $K_a \geq V_p \Delta t$, where Δt is the time step of the controller.

5.3. Optimal reciprocal collision avoidance

For each robot, given a preferred velocity $\mathbf{v}_{pref_i}^k$ and the current velocities and positions of its neighbors, a collision-free velocity $\mathbf{v}_{cf_i}^k$ is computed. In order to avoid collisions while guaranteeing smooth motions, a local optimal reciprocal collision avoidance in velocity space (ORCA) based on velocity obstacles (Fiorini and Shiller 1998) is used, which exploits the fact that all controlled robots in the environment react following the same scheme and thus avoids oscillations. This method guarantees oscillation-free and smooth motions in multi-robot scenarios and thus is very well suited for image display. ORCA was presented by van den Berg et al. (2009) for holonomic robots and in the following an overview is given.

Consider two holonomic robots i and j of radius r_i and r_j at positions $\mathbf{p}_i$ and $\mathbf{p}_j$ and subject to current velocities $\mathbf{v}_{cf_i}^{k-1}$ and $\mathbf{v}_{cf_j}^{k-1}$. The velocity obstacle for robot i induced by robot j is defined as the set of relative velocities $\bar{\mathbf{v}} = \mathbf{v}_i - \mathbf{v}_j$ leading to collision

$$VO_{i|j}^\tau = \left\{\bar{\mathbf{v}} \mid \exists t \in [0, \tau],\ t \cdot \bar{\mathbf{v}} \in D(\mathbf{p}_j - \mathbf{p}_i,\ r_i + r_j)\right\}, \quad (10)$$

with $D(\mathbf{p}, r) = \{\mathbf{q} \mid \|\mathbf{q} - \mathbf{p}\| < r\}$ the open ball of radius r. The set of collision-free velocities $ORCA_{i|j}^\tau$ for robot i with respect to robot j can geometrically be constructed from $VO_{i|j}^\tau$. First, the minimum change in velocity that needs to be added to $\bar{\mathbf{v}}$ to avoid a collision,

$$\mathbf{u} = \left(\underset{\bar{\mathbf{v}} \in \partial VO_{i|j}^\tau}{argmin} \|\bar{\mathbf{v}} - (\mathbf{v}_i^{opt} - \mathbf{v}_j^{opt})\|\right) - (\mathbf{v}_i^{opt} - \mathbf{v}_j^{opt}), \quad (11)$$

is computed, where $\mathbf{v}_i^{opt}$ is the optimization velocity, set to the current velocity $\mathbf{v}_{cf_i}^{k-1}$ of the robot. Then

$$ORCA_{i|j}^\tau = \{\mathbf{v}_i | (\mathbf{v}_i - (\mathbf{v}_i^{opt} + c\mathbf{u})) \cdot \mathbf{n} \geq 0\}, \quad (12)$$

follows, where $\mathbf{n}$ denotes the outward normal of the boundary of $VO_{i|j}^\tau$ at $(\mathbf{v}_i^{opt} - \mathbf{v}_j^{opt}) + \mathbf{u}$, and c defines how much each robot gets involved in avoiding a collision. Here $c = \frac{1}{2}$ means both robots i and j help to equal amounts to avoid colliding with each other; $c = 1$ means robot i fully avoids collisions with a dynamic obstacle j. Likewise, the velocity obstacle can also be computed for static obstacles (van den Berg et al. 2009).

The set of collision-free velocities in horizon τ_i for robot R_i, $ORCA_i^\tau$ is given by

$$ORCA_i^\tau = S_{AHV_i} \cap \bigcap_{j \neq i} ORCA_{i|j}^\tau, \quad (13)$$

where S_{AHV_i} is the set of allowed holonomic velocities and for holonomic robots $S_{AHV_i} = D(\mathbf{0}, V_{H_i}^{max})$, a disk with radius the maximum velocity of the robot.

The optimal collision-free velocity for robot i is given by

$$\mathbf{v}_{cf_i}^k = \underset{\mathbf{v}_i \in ORCA_i^\tau}{argmin} \|\mathbf{v}_i - \mathbf{v}_i^{pref}\|. \quad (14)$$

This is an optimization with linear constraints which can be solved efficiently. If the optimization is unfeasible, the time horizon is decreased for that time step until the problem becomes feasible. This is a 3D linear program which is always feasible and that can be solved by a randomized algorithm running in $\mathcal{O}(N_{n,i})$ expected time for each robot i, where $N_{n,i}$ is the number of neighboring robots. In our case, if the time horizon goes below a given threshold the robot is stopped for that time step.

5.4. Extension to non-holonomic kinematics

The kinematic constraints of differentially driven robots, like those in the experiments of this work, are taken into account by Alonso-Mora et al. (2010), which builds on the ORCA method for holonomic robots (van den Berg et al. 2009). This extension, like ORCA, guarantees oscillation-free and smooth motions in multi-robot scenarios and thus is very well suited for image display. A brief introduction of this method is presented here. In particular, a differentially driven robot is considered that tracks a holonomic trajectory given by a velocity vector. By limiting the set of velocity vectors and trajectories of the differentially driven robot it can be guaranteed that tracking is achieved within a fixed maximum error $\mathcal{E}_i > 0$.

Consider known $\mathcal{E}_i$ for each robot i which guarantees for every pair of robots that there is no overlap of the extended radius, $(r_i + \mathcal{E}_i) + (r_j + \mathcal{E}_j) \leq \|\mathbf{p}_i^k - \mathbf{p}_j^k\|$, where r_i, r_j represent the radius of robots i, j. This is achieved by having a desired value of the maximum error and decreasing it stepwise when robots are close to each other so that the previous equation holds.

In this case, S_{AHV_i} represents the set of holonomic velocities for which tracking within $\mathcal{E}_i$ error is guaranteed and $ORCA_{i|j}^\tau$ is then the set of collision-free velocities for horizon τ computed for a holonomic robot at position $\mathbf{p}_i^k$,

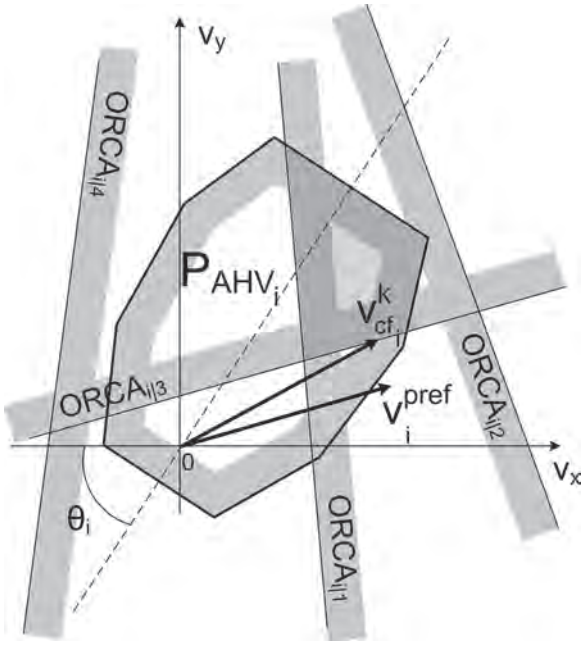

Fig. 5. Optimization in velocity space for a differentially driven robot with current orientation θ_i in a scenario with five robots. The optimal collision-free velocity is found within the linear constraints and minimizing the distance to the preferred velocity.

radius $r_i + \mathcal{E}_i$ and current velocity $\mathbf{v}_{cf_i}^{k-1}$ with respect to a holonomic robot at position $\mathbf{p}_j^k$, radius $r_j + \mathcal{E}_j$ and current velocity $\mathbf{v}_{cf_j}^{k-1}$. S_{AHV_i} was derived in closed form by Alonso-Mora et al. (2010). Figure 5 shows the set $ORCA_i^\tau$ for a configuration with multiple robots, where S_{AHV_i} is approximated by the convex polygon P_{AHV_i}. In case of unfeasibility of the optimization, the symmetrical mirrored polygon to P_{AHV_i} within S_{AHV_i} is used. This represents a movement in opposite direction to that of the desired goal.

Finally, $\mathbf{v}_{cf_i}^k$ is mapped to the corresponding control inputs, which guarantee collision-free motion. The optimal controls are given by $\omega = \min(\frac{\theta_H}{T}, \omega_{\max})$ and $v = \min(v_{\mathcal{E}}^*, v_{\max} - |\omega(t)|\frac{l_w}{2})$, where $v_{\mathcal{E}}^*$ is the linear velocity that minimizes the tracking error for $\omega = \frac{\theta_H}{T}$ as defined in Alonso-Mora et al. (2010), θ_H is the angular difference between $\mathbf{v}_{cf_i}^k$ and the current orientation of the robot, T is a constant defining the time to achieve the desired orientation, l_w is the inter-wheel distance and $v_{\max}$ and $\omega_{\max}$ are the maximum angular and linear velocities.

The ORCA method was also extended to car-like robots by Alonso-Mora et al. (2012) and can likewise be applied in this framework.

5.5. Theoretical guarantees

In this section theoretical guarantees on collision-free trajectories and convergence are given. First, the case of point robots is studied to give an insight on the ideal trajectories. Second, the case of disk robots is discussed and a method to solve deadlocks is presented, which in practice leads to convergence.

5.5.1. Definitions Denote by Σ_I the set of permutations of elements of I and $\hat{\Sigma}_k$ the set of goal assignment functions at iteration k. Note that the permutation function $\sigma_k \in \Sigma_I$, is intrinsically independent of the iteration, whilst the assignment function $\hat{\sigma}_k \in \hat{\Sigma}_k \subset \hat{\Sigma} = \bigcup_{n\in\mathbb{N}} \hat{\Sigma}_n$, clearly depends on it. A functional F linking the permutation functions with the assignment functions at a given time is defined as

$$\begin{aligned} F: \quad \Sigma_I \times \mathbb{N} &\longrightarrow \quad \hat{\Sigma} \\ (\sigma_k,\, r) &\longmapsto \quad \hat{\sigma}_r, \end{aligned} \tag{15}$$

where $\sigma_r = \sigma_k \in \Sigma_I$ is the permutation function associated with the assignment function $\hat{\sigma}_r$, and which is equal to that associated with $\hat{\sigma}_k$ at time step k, thus

$$\hat{\sigma}_r = F(\sigma_k, r) \Leftrightarrow \sigma_r = \sigma_k. \tag{16}$$

5.5.2. Collision-free trajectories The local collision avoidance method guarantees collision-free motions, produces smooth trajectories and performs well in crowded scenarios. Detailed proofs were given by van den Berg et al. (2009) for holonomic robots, Alonso-Mora et al. (2010) for differentially driven robots and Alonso-Mora et al. (2012) for car-like robots.

5.5.3. Convergence for holonomic point robots The control algorithm given by Equations (5), (7) and (9) guarantees, in the case of holonomic point robots, collision-free trajectories and convergence to the goal positions. Moreover, trajectories are oscillation-free in the noise-free case, that is to say, there is no goal reassignment.

Theorem 1 (Collision-free trajectories)**.** *Trajectories of holonomic point robots are collision-free for the limit case* $\Delta t \to 0$*, velocity following Equation* (9) *and with no local collision avoidance.*

Lemma 1 (Intersecting paths)**.** *Without loss of generality consider* $\boldsymbol{p}_1 = (-a, 0)$ *and* $\boldsymbol{p}_2 = (a, 0)$ *the position of two robots;* $\boldsymbol{g}_1 = (x, y)$ *and* $\boldsymbol{g}_2 = (r, s)$ *the respective goals. If* $r \geq 0$ *and* $s \geq 0$*, the paths of the robots cross if and only if*

$$x \in (a, r), \quad y \in \left[0, s\frac{x-a}{r-a}\right]. \tag{17}$$

For $r < 0$ *or* $s < 0$*, the previous equations extend by symmetry.*

Proof. Consider, without loss of generality two robots in positions $\mathbf{p}_1 = (-a, 0)$ and $\mathbf{p}_2 = (a, 0)$, where $a \in \mathbb{R}_+$. Let $\mathbf{g}_1 = (x, y) \in \mathbb{R}^2$ and $\mathbf{g}_2 = (r, s) \in \mathbb{R}^2$ their respective assigned goals. See Figure 6. Owing to symmetry let us consider the case $r \geq 0$ and $s \geq 0$.

First note that paths given by the controller defined by Equation (9) are straight lines between the current position and the assigned goal.

From optimality of the assignment, the paths verify

$$\|\mathbf{g}_1 - \mathbf{p}_1\|^2 + \|\mathbf{g}_2 - \mathbf{p}_2\|^2 < \|\mathbf{g}_2 - \mathbf{p}_1\|^2 + \|\mathbf{g}_1 - \mathbf{p}_2\|^2 \quad (18)$$

which directly implies $x < r$.

If $y < 0$ or $x \leq a$ paths are trivially non-crossing. If $r > x > a$ then, from simple trigonometry paths cross if and only if $0 \leq y \leq s\frac{x-a}{r-a}$. □

Proof of Theorem 1. Assume the two goals are not in the same position.

Collision-free trajectories derive from the fact that, in the case of crossing paths, both robots are not simultaneously in the intersection.

Without loss of generality consider two robots and two goal positions as defined in Lemma 1, where the paths are intersecting. Moreover, consider that the goal assignment remains constant.

Define $\overline{\mathbf{pq}}$ the segment given by positions $\mathbf{p}$, $\mathbf{q}$. From Equation (17) and simple geometry it is obtained that

$$\|\overline{\mathbf{p}_1\mathbf{O}}\| > \|\overline{\mathbf{p}_2\mathbf{O}}\| \text{ and } \|\overline{\mathbf{O}\mathbf{g}_1}\| < \|\overline{\mathbf{O}\mathbf{g}_2}\|, \quad (19)$$

where $\mathbf{O} \in \Re^2$ represents the intersection of segments $\overline{\mathbf{p}_1\mathbf{g}_1}$ and $\overline{\mathbf{p}_2\mathbf{g}_2}$. See Figure 6.

Let us parametrize the segments $\overline{\mathbf{p}_1\mathbf{O}}$ and $\overline{\mathbf{p}_2\mathbf{O}}$ by the distance to the intersection $\mathbf{O}$. Denote $d_{1,O} = \|\overline{\mathbf{p}_1\mathbf{O}}\|$ and $d_{2,O} = \|\overline{\mathbf{p}_2\mathbf{O}}\|$ the length of the segments. Further denote by $v_1(e)$ and $v_2(e)$ the speed of robots 1 and 2 at a distance e from $\mathbf{O}$, where the speed follows Equation (9).

From $\|\overline{\mathbf{O}\mathbf{g}_1}\| < \|\overline{\mathbf{O}\mathbf{g}_2}\|$ and (9) it is obtained that $v_1(0) \leq v_2(0)$, which due to the monotonicity of the velocity implies

$$v_1(e) \leq v_2(e), \quad e \in [0, d_{2,O}]. \quad (20)$$

Finally the times at which robots 1 and 2 reach the intersecting point are related by

$$\begin{aligned} t_{\mathbf{p}_1,\mathbf{O}} &= \int_0^{d_{1,O}} \frac{1}{v_1(e)}\, de > \int_0^{d_{2,O}} \frac{1}{v_1(e)}\, de \\ &\geq \int_0^{d_{2,O}} \frac{1}{v_2(e)}\, de = t_{\mathbf{p}_2,\mathbf{O}}, \end{aligned} \quad (21)$$

where the first inequality holds from Equation (19) and $v_1(e) \geq 0$. The second inequality holds from Equation (20).

This proves that for the case of no goal reassignment trajectories are collision-free. The proof extends likewise to the case of reassignment of goal positions by considering the current positions as start positions. □

Theorem 2 (Convergence)**.** *Convergence to the set of goal positions is guaranteed for holonomic point robots.*

Proof. In seek of clarity, consider the abuse of notation $j = \sigma_k^*(i)$ for robot i. This is goal $\mathbf{g}_j$ is assigned to robot i.

From Equation (9), in each step every robot moves in a straight line towards its assigned goal.

$$\mathbf{v}_{pref,i}^k = K_c(\mathbf{g}_j - \mathbf{p}_i^k), \quad \forall i \in I, \quad (22)$$

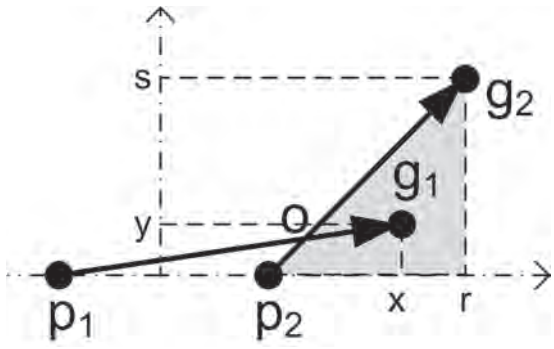

Fig. 6. Optimal goal assignment for two robots.

where $K_c > 0$ is obtained from Equation (9) and verifies $K_c\Delta t \leq 1$. Therefore, the new position of each robot is given by

$$\begin{aligned} \mathbf{p}_i^{k+1} &= \mathbf{p}_i^k + \mathbf{v}_{pref,i}^k \Delta t = \\ &= \mathbf{p}_i^k + K_c\Delta t(\mathbf{g}_j - \mathbf{p}_i^k), \quad \forall i \in I. \end{aligned} \quad (23)$$

This implies that $\mathbf{g}_j - \mathbf{p}_i^{k+1}$ and $\mathbf{g}_j - \mathbf{p}_i^k$ are collinear vectors and $\|\mathbf{p}_i^k - \mathbf{g}_j\|^2 \geq \|\mathbf{p}_i^{k+1} - \mathbf{g}_j\|^2$. Therefore, using the cost defined in Equation (7), the cost of the current assignment is reduced

$$C(F(\sigma_k^*, k+1)) \leq C(\hat{\sigma}_k^*). \quad (24)$$

The cost of the optimal assignment at iteration $k+1$ verifies then

$$0 \leq C(\hat{\sigma}_{k+1}^*) \leq C(F(\sigma_k^*, k+1)), \quad (25)$$

which implies that the cost of the goal assignment monotonically decreases with time and is upper bounded by the case where the goal assignment permutation at time step $k+1$ is equal to that at time step k, conversely $\sigma_{k+1}^* = \sigma_k^*$.

For the upper bounding case of Equation (25), convergence to the set of goal positions is guaranteed by Equation (23) and Theorem 1. From Equation (25), convergence in the general case is thus guaranteed

$$0 \leq \lim_{k\to\infty} C(\hat{\sigma}_k^*) \leq \lim_{k\to\infty} C(F(\sigma_0^*, k)) = 0. \quad (26)$$

□

5.5.4. Convergence for disk robots If all of the goals are at a distance greater than twice the diameter of a robot, accessibility and thus convergence is guaranteed. In the general case, where the distance is at least twice the radius of a robot, for disk robots convergence is not guaranteed. Nevertheless, from Theorem 1 ideal trajectories (straight line paths without collision avoidance) reduce the likelihood of frontal collisions and therefore of deadlocks. Moreover, the choice of the cost function following Equation (7) leads to improved convergence in crowded scenarios, as shown subsequently.

In Figure 7, a situation with three robots is shown to illustrate how the goal assignment based on the chosen cost

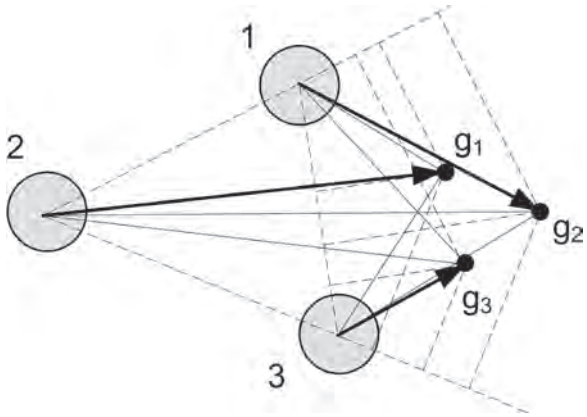

Fig. 7. Goal assignment for three robots minimizing the sum of squared distances. The geometric construction of the optimal goal assignment is displayed. Trajectories are collision-free.

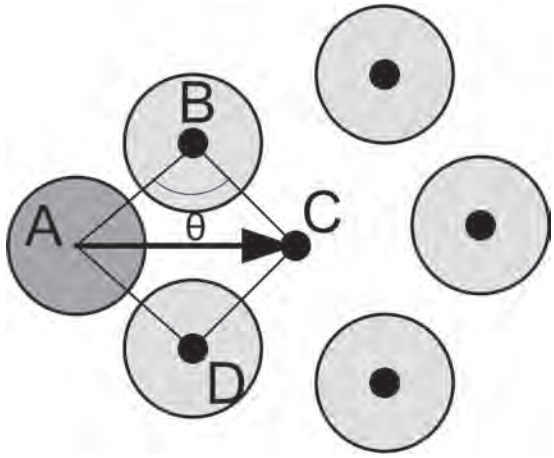

Fig. 8. Example with six robots, represented by a disk, and six goal positions, represented by a thick black dot. Five of the robots (light gray) are situated on goal positions and one (dark gray) remains to converge. If $\widehat{\mathrm{ABC}} > \frac{\pi}{2}$ a goal reassignment A–B and B–C is optimal and the deadlock is avoided. Otherwise, a deadlock situation arises when robot A stops.

function helps in avoiding deadlock situations. The goal assignment as given by Figure 7 leads to convergence avoiding deadlocks. With the help of the dashed lines and Lemma 1 the optimal goal assignment can be visually computed. A cost function defined by the sum of distances to the goal leads to a deadlock in the same scenario, where robots 1–3 are assigned to goal positions g_1, g_2, g_3, respectively, and a deadlock situation arises finally, where robot 2 may not reach its goal. Nevertheless, the local collision avoidance method would, in this case, deviate the robots from their desired trajectories to finally achieve convergence.

A similar, more extreme, situation appears in highly crowded environments near convergence, such as that shown in Figure 8, where the light gray robots are at their goal positions and the dark gray robot has point C as goal position. In this case, as represented in Figure 8, using the sum of distances the current goal assignment remains optimal, and therefore the robot at position A will never reach goal position C. On the other hand, using the sum of squared distances from Equation (7) as cost function, a reassignment of the goal positions between the robots at positions A and B (or A and D) happens whenever the angle $\theta = \widehat{\mathrm{ABC}}$ exceeds $\frac{\pi}{2}$. (Here $\overline{\mathrm{AC}}^2 = \overline{\mathrm{AB}}^2 + \overline{\mathrm{BC}}^2 - 2\overline{\mathrm{AB}}\,\overline{\mathrm{BC}}\cos\theta \leq \overline{\mathrm{AB}}^2 + \overline{\mathrm{BC}}^2 \quad \Leftrightarrow \quad \theta \in [0, \frac{\pi}{2}]$.) This is the case when the distance AC is greater than a certain threshold. If $\theta \leq \frac{\pi}{2}$, a deadlock situation is obtained. In this case, a movement of the robot at position A around either robots at positions B or D would lead to an increase in the angle implying a goal reassignment, which would also solve the deadlock. In the following subsection the method implemented for solving this situations is explained.

5.5.5. Deadlocks Note that convergence is not strictly guaranteed by the given method for disk robots if the distance between goal positions is lower than $4r$, with r the radius of the robots, and in the presence of measurement noise and suboptimal assignments. In very rare cases and highly packed scenarios with a large number of robots, deadlock situations close to convergence have been observed. In all cases it implied one single robot whose path to the goal was blocked by other robots already in their final goal positions. This falls in the case shown in Figure 8 where $\widehat{\mathrm{ABC}} < \frac{\pi}{2}$ and $\widehat{\mathrm{ADC}} < \frac{\pi}{2}$ and reassignment is not optimal. In practice, this might happen also for slightly greater angles due to the suboptimality of the goal assignment.

These deadlocks can be detected by comparing the preferred velocity of the robot with its collision-free velocity. If the difference is above a certain threshold during N_d iterations, then the robot is considered to be in a deadlock situation. Moreover, this situation is only checked for robots near their goal position, where deadlocks are possible.

If a robot is found to be in a deadlock situation, the cost of it being assigned to its current goal is increased by a fixed value C_d. This increase in cost is then kept for the remaining iterations and implies a reassignment of the robots to the goal positions which solves the deadlock situation.

6. Animation display

The previous sections described a method to display a single image with multiple mobile robots. This section extends the method to display animations specified by a sequence of input images or *keyframes*. Modifications of the methods of Sections 4 and 5 to achieve smooth animation are described below.

6.1. Goal generation

For each image (frame) I_f, $f \in [1, N_F] \subset \mathbb{N}$, a set of goal positions $\mathcal{G}_f$ is obtained using the method in Section

Fig. 9. Goal positions (black circles) for 100 robots representing the leaf of the animation of Section 7. The initial samples (blue stars) were obtained from the previous keyframe, using an estimate of the translation and rotation of the global shape.

4, where N_F is the number of keyframes. In parts of the animation where the number of regions remains constant and there is sufficient correlation, the goal positions computed for region R_i in the previous frame $\mathcal{G}_{f-1}$ serve to initialize the computation of the goal positions of region R_i in the current frame $\mathcal{G}_f$. The translation of each region R_i is computed along frames using its centroid, while the rotation is computed along frames by matching of features (Canny edges) on the region border. This provides an estimate of the translation T_i^f and rotation M_i^f of each region R_i for every pair of consecutive frames. The initial samples for the optimization of Section 4 are computed by applying T_i^f and M_i^f to $\mathcal{G}_{f-1}$ (independently for each region R_i). This initialization reduces computation time and disparities between consecutive goal sets. In Figure 9 the goal positions (black circles) for 100 robots representing the leaf image of the animation of Section 7 are displayed. The initial samples (blue stars) were obtained by translation and rotation of the goal positions found for the previous keyframe.

6.2. Real-time control

The controller of Section 5 drives the robots through the given goal positions, representing the frames in sequence at given time instances with $K_f \Delta t$ separation, where $K_f \in \mathbb{N}$ is a design constant and Δt is the time step of the controller. Therefore, the set of goals is kept constant during K_f steps of the controller. In order to achieve smoother motions, the velocities of the robots are synchronized when the set of goal positions is changed, by selecting

$$\mathbf{v}_{pref_i}^k = \min\left(v_{\max}, \frac{\|\mathbf{g}_{\sigma_k^*(i)} - \mathbf{p}_i^k\|}{K_f \Delta t}\right) \frac{\mathbf{g}_{\sigma_k^*(i)} - \mathbf{p}_i^k}{\|\mathbf{g}_{\sigma_k^*(i)} - \mathbf{p}_i^k\|}, \quad (27)$$

as the preferred velocity (substituting Equation (9)) and keeping it constant for K_f iterations. The constant K_f is directly related to the distance between consecutive frames and inversely related to the maximum speed of the robots.

With this method, an artist who creates input images only needs to specify keyframes for an animation, with no requirement to provide a dense time sampling in the input. Appropriate motion of the robot pixels between keyframes is generated by the system itself.

7. Experimental evaluation

This section presents experimental results for two physical platforms, a group of 14 modified e-puck robots (Mondada et al. 2009) and a group of 50 Elisa robots (Alonso-Mora et al. 2011a), and for simulations with 1,000 robots. Section 7.1 contains an overview of the experimental setup. To analyze system performance, Sections 7.2, 7.3 and 7.4 describe experiments with the e-puck robots. To demonstrate the scalability of the system in terms of robot coordination and computational complexity, Sections 7.2 and 7.3 contain experiments with simulations for 1,000 robots, and Section 7.5 contains experiments with 50 Elisa robots. The videos accompanying this section are available in Extensions 1 and 2.

7.1. Experimental setup

The experimental setup is shown in Figure 10. The two physical robots, both differential drive, are shown in Figure 11. The parameters of the modified e-puck robot are given in Table 1. The e-puck is equipped with an array of 3×3 RGB LEDs but, for these experiments, all nine elements of the LED array are set to the same color. The Elisa robot is described in Section 7.5.1 and has one RGB LED.

The central computer sends control signals (wheel speeds, RGB color, and IR LED commands) to the robots using radio communication at 10 Hz. The central radio module has a 2.4 GHz transceiver and sends messages by unicast with a special embedded baseband protocol engine for the packet based data link layer, to provide low-power, high-performance communication acknowledgement and retransmission capabilities. This protocol supports a 10 Hz update rate for up to 100 robots.

The robots are tracked using an overhead camera which is 2.3 m above the deployment plane. The camera is a Point Grey Flea2 color camera with $1{,}600 \times 1{,}200$ resolution, running at a frame rate of 10 Hz. The infrared filter is removed to allow the detection of both visual and infrared light. Robots are localized with a precision of 2 mm and 3°.

Table 1. Description of the modified e-puck robot.

	Value	Units	Description
l_w	0.0525	m	Distance between wheels
d_w	0.041	m	Diameter of the wheels
d_A	0.09	m	Robot's maximum diameter
$v_{\max}$	0.13	m s^{-1}	Maximum linear speed
$\omega_{\max}$	4.96	rad s^{-1}	Maximum angular speed
$\epsilon_{\max}$	0.01	m	Maximum extension of the robot's radius
τ	2	s	Time to collision horizon
T_o	0.35	s	Time to achieve desired orientation
V_p	0.12	m s^{-1}	Preferred speed
K_a	0.1	m	Linear speed reduction factor
ϵ_A	0.01	m	Maximum goal assignment error

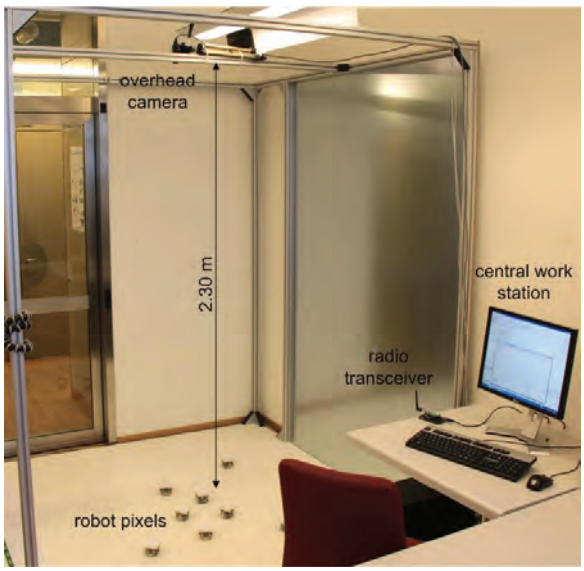

Fig. 10. The system consists of a group of differentially driven robot pixels, an overhead camera, and a standard PC with radio module.

Fig. 11. Robot pixels. At left, the modified e-puck robot with a 3×3 RGB LED array. At right, the Elisa robot with a single RGB LED (see Section 7.5.1). The blue spots are infrared LEDs.

Each e-puck is equipped with a set of six infrared LEDs which emit a unique code used for identification and tracking. This approach does not extend to large robot swarms because the number of LEDs imposes a limit on the number of available codes. Hence, the Elisa robots are tracked instead by using a more sophisticated vision system.

The simulations with a group of 1,000 robots are carried out identically to the physical experiments, except that the tracking is omitted. The vehicle kinematics are simulated using actuation noise comparable to the real system.

7.2. Image display

This section contains a quantitative analysis of system performance. For these experiments, we chose a set of simple yet diverse images. They include patterns of a disk and a line (two disconnected convex patterns within the same image), and an L-shape and T-shape (non-convex patterns). These images provide a benchmark to analyze image formation when there is varying pixel resolution, multiple patterns within an image, and non-convexity of patterns. Figure 12 top row shows sample trajectory traces of a subset of the robots from a single experimental run. Figure 12 middle row shows the images displayed with 14 e-puck robots. Figure 12 bottom row shows the images displayed with 1,000 simulated robots. In general, we found that qualitative and quantitative results obtained from simulations with 1,000 robots are comparable to the results achieved with the 14 e-puck robots. The accompanying video materials contain representative runs of the physical experiments and simulations.

7.2.1. Overall performance Visually, for both the physical and simulated experiments depicted in Figure 12, transitions appear smooth despite frequent interactions between robots. However, from a quantitative point of view, we are interested in the repeatability of the resulting motion. To this end we draw statistics over 25 experimental runs with 14 e-puck robots forming the images of Figure 12. The goal

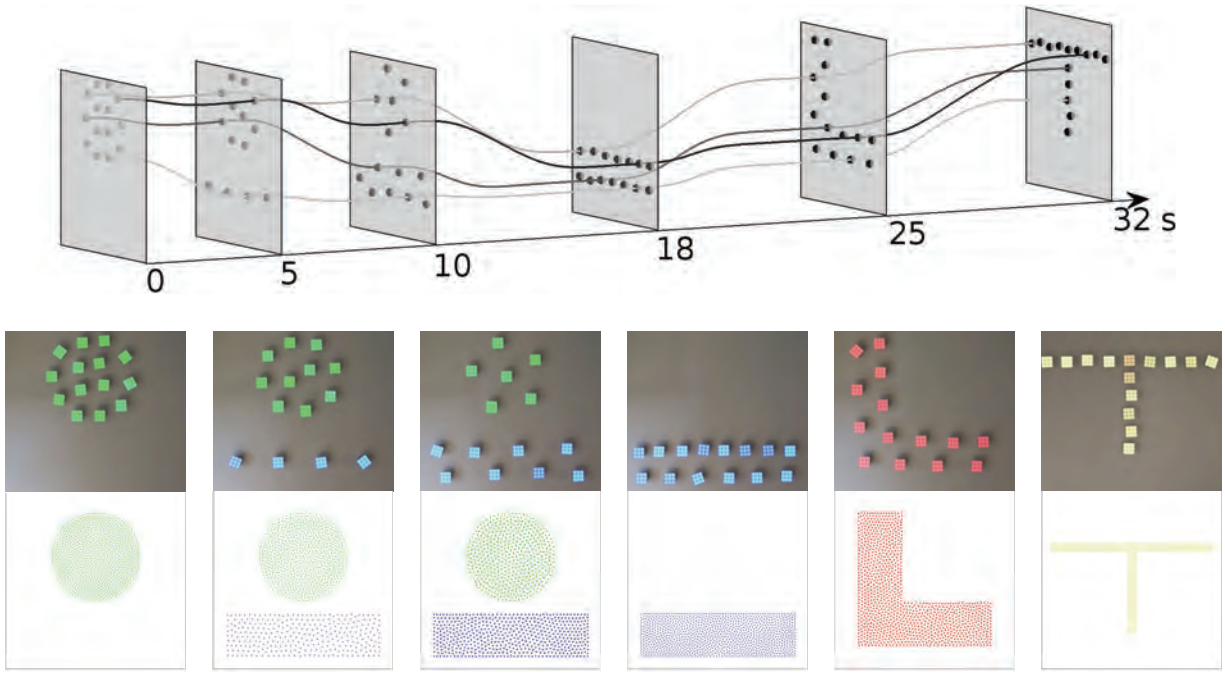

Fig. 12. *Top:* Temporal visualization of transitions between patterns, with traces of four robot trajectories to illustrate motion patterns. *Middle:* Set of static images for the experiments with 14 e-puck robots. The robots pass in chronological order through a disk, a disk and a line (10 robot disk and 4 robot line), a disk and a line (6 robot disk and 8 robot line), a line, an L-shape pattern, and a T-shape pattern. *Bottom:* The same sequence of static images with 1,000 simulated robots. The full video is available in Extension 1.

positions to represent the images are regenerated for each of the runs anew and therefore vary slightly from run to run (see Section 4.5 for the study of the variation in goal distributions due to local minima). In this set of experiments the robots always start from the same grid configuration. In Figure 13 we report on the repeatability between the experimental runs in terms of total path length. For each box, the central mark is the median, the edges of the box are the 25th and 75th percentiles, the whiskers extend to the most extreme data points not considered outliers, and outliers are plotted individually. Each column represents the variability in path length of a single robot over all experimental runs for identical initial robot positions (left *y*-axis). The right-most box represents the cumulative path length over 14 robots and its variability taken over all experimental runs (right *y*-axis). We observe two key points: first, variation in path length between different runs is substantial for an individual robot. We attribute this to the local convergence of the goal generation algorithm, which results in the same robot being assigned to different positions within the ensuing image patterns over different runs. Different distributions of goal positions for the same pattern were shown in Figure 4. Second, the overall variation in summed total path length over all robots is small, indicating that the goal assignment successfully handles the small perturbations in the system as well as the variability in the goal distributions to obtain fast convergence to all images.

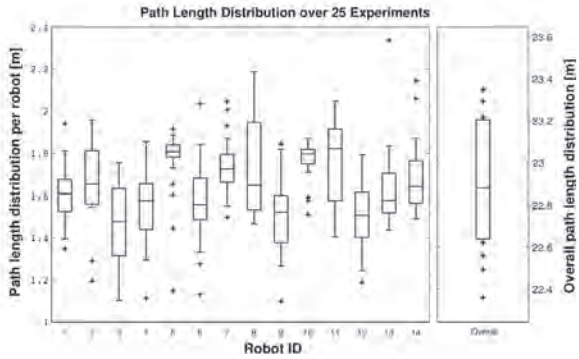

Fig. 13. *Left:* Variation in path length between different runs of the same experiment. Each column represents the variability in path length over all experimental runs for a single robot always starting from the same initial position (left *y*-axis). *Right:* Summed total path length and its variability over all experiments (right *y*-axis).

We note again that our algorithm is not designed to distribute target locations in such a way that all robots reach their destination simultaneously. It rather minimizes the summed squared traveled distance over all robots. In fact, by minimizing the sum of squared distances to the goals, long paths are penalized and therefore avoided. Nevertheless, synchronization could be achieved by specifying a

desired time to form the pattern and subsequently adjusting the preferred velocities of the individual robots.

7.2.2. Analysis of suboptimality in goal assignment In order to gain a better understanding of how the goal assignment algorithm performs against limited computational power or limited knowledge present in the system, further experiments are conducted for evaluating the effect of the parameter ϵ_A in the goal assignment. As detailed in Section 5.1 the parameter ϵ_A represents a bound in the error committed by the auction-based goal assignment for each individual robot with respect to the optimal assignment. We show that our algorithms perform properly even in cases of high suboptimality of the goal assignment. In particular, the sequence of patterns represented in Figure 12 is repeated 25 times with the 14 e-puck robots for values of ϵ_A of 1, 0.1, 0.01 and 0.001 m^2 and the results are analyzed.

In the left image of Figure 14 the computation time of the goal assignment is analyzed. Each column represents the variability over all iterations (10 times per second) and 25 experimental runs for a given value of the parameter ϵ_A. For $\epsilon_A = 1$ m^2 very few iterations are needed to find an assignment (this is most of the time the initial guess) with individual error below $\epsilon_A = 1$ m^2, but the found solution is extremely suboptimal, as observed in the left image of Figure 15. For $\epsilon_A \leq 0.1$ m^2 the worst-case computation increases inversely to ϵ_A when no prior information is available (in the case of changing to a new image). On the other hand, by computing a more accurate assignment, the assignment remains optimal for a longer time and the computation in the following steps benefits from the prior information, thus the mean time decreases with ϵ_A again.

In the right image of Figure 14 the execution time (time to transform between all images) of the experiment is analyzed. Each column represents the variability over all 25 experimental runs for a given value of the parameter ϵ_A. With a decrease of ϵ_A, the error due to suboptimality in the goal assignment is reduced, thus producing shorter paths and faster execution times. For high values of ϵ_A ($\epsilon_A = 1$ m^2) high variability is observed, this is due to the extreme suboptimality of the computed goal assignment resulting in very different paths. For low values of ϵ_A the variability arises from small differences in the paths due to the local collision avoidance and to inaccuracies in the system, producing slightly different motions in each experiment. In the experiments no visual difference was observed for the cases $\epsilon_A = 0.01$ and 0.001 m^2 when compared with the optimal assignment.

Finally, in Figure 15 an example of the trajectories of the 14 e-puck robots transforming between two successive images of the sequence is presented for $\epsilon_A = 1$ m^2 (left) and $\epsilon_A = 0.001$ m^2 (right). For $\epsilon_A = 0.001$ m^2 an assignment equal to the optimal one is obtained and thus the trajectories are short, smooth and oscillation-free. In this case, the assignment error is below the physical radius of the robot and the separation between goals, guaranteeing that the optimal goal assignment is obtained. On the other hand, for $\epsilon_A = 1$ m^2 extremely suboptimal assignments result leading to long and crossing trajectories and sudden changes in direction due to reassignments. Although the goal assignment being suboptimal in this case is expected, the overall system and the collision avoidance algorithm in particular proved to be robust and worked successfully even under bad assignment. Furthermore, collision-free trajectories are obtained in all of our experiments, independent of the number of robots or the shape of images. Further experiments evaluating our local collision avoidance method are given in Alonso-Mora et al. (2010).

7.3. Animation display

This section contains two experiments, one with physical robots and one in simulation, to demonstrate animation display. In the first experiment, 14 e-puck robots show an animation of a walking human, which falls into a heap, which turns into a flower circled by a bee, as shown in Figure 16. The required input is a small number of keyframes to provide the required appearance at the major time steps in the animation. In between keyframes, the robots are guided solely by the system algorithms. This experiment demonstrates that an animated story can be achieved even with a small number of robot pixels. Audio (for example, buzzing of the bee) provides a helpful extra cue for the viewer.

The second experiment demonstrates system scalability. A total of 1,000 robots in simulation show a leaf from growing in spring, to withering in fall, and finally being blown from the branch. The experiment demonstrates that a group of 1,000 robots can closely resemble input artwork, with shape and contours of objects defined in much more detail, although some transitions (such as fast rotations) cause distortions in the leaf boundary due to optimal reassignments of robots.

7.4. Extensions

This section describes three extensions to image and animation display. (1) Dynamic image display in which the target image undergoes translation, rotation, and scaling. (2) Handling dynamic obstacles for both image and animation display, in which mobile objects that are not under control of the system affect the robot pixels as they move through the workspace. (3) Perturbation recovery for both image and animation display, in which a user moves one or more robot pixels by hand and the system recovers the target configuration.

7.4.1. Dynamic image display In this case, the desired image remains constant over time but is subject to a translation, rotation and isometric deformation. The transformation can be represented by three additive factors: a translational motion of the center of the pattern $\mathbf{p}_{OG}$

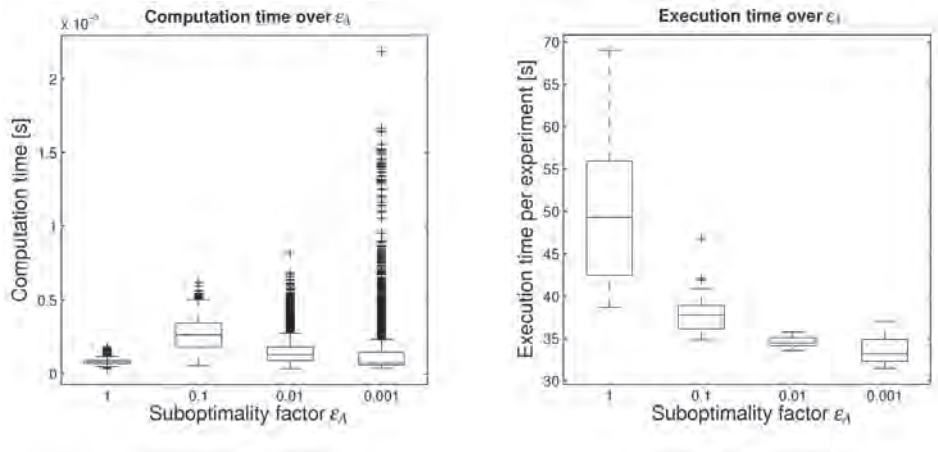

Fig. 14. *Left:* Variation in computation time of the goal assignment. Each column represents the variability over all iterations and experiments for a given value of the parameter ϵ_A. *Right:* Variation in execution time of the experiment (time to transform between all images). Each column represents the variability over all experiments for a given value of the parameter ϵ_A.

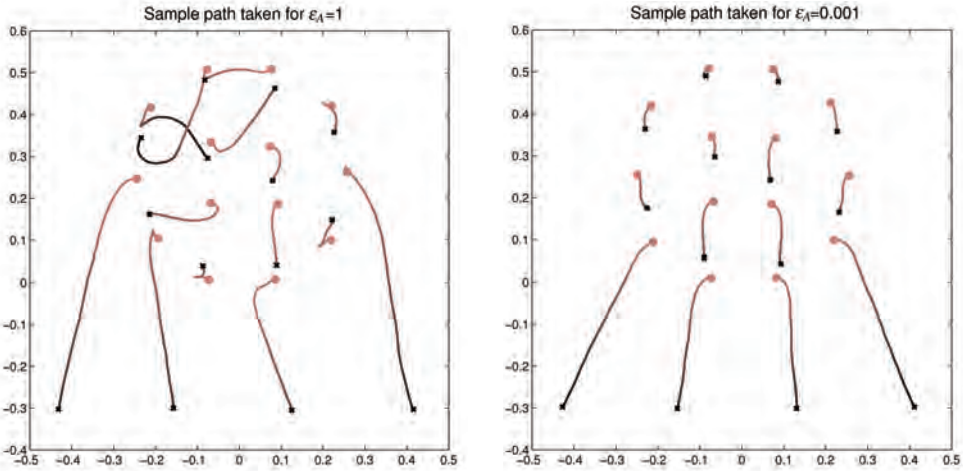

Fig. 15. Trajectories for 14 e-puck robots in the transition from a disk to a disk (10 robots) and a line (4 robots). Start and final positions are represented with red circles and black crosses, respectively. *Left:* Suboptimal goal assignment with $\epsilon_A = 1$ m^2. *Right:* Optimal goal assignment with $\epsilon_A = 0.001$ m^2.

expressed by a linear velocity $\mathbf{v}_G$; a rotation around $\mathbf{p}_{OG}$ given by an angular velocity ω_G; and an isometric deformation factor K_{eG} which is positive for expansion and negative for contraction of the pattern.

The set of goal positions is computed following the method of Section 4. This computation needs to be run only once for the input image. In subsequent time instances the goal positions undergo the same transformation as the entire image. Assume that the given pattern is represented at start time $t = 0$ through the set of goal positions $\mathcal{G}^0 = [\mathbf{g}_1^0, \ldots, \mathbf{g}_N^0]$, where $\mathbf{g}_j^0 \in \mathbb{R}^2$, as obtained from the initial image. At time t, the set of goal positions is represented by $\mathcal{G}^t = [\mathbf{g}_1^t, \ldots, \mathbf{g}_N^t]$, where $\mathbf{g}_j^t \in \mathbb{R}^2$ and can be computed directly from the initial goal positions $\mathcal{G}^0$ for the case of constant motion and deformation by

$$\mathbf{g}_j^t = \mathbf{p}_{OG} + t\mathbf{v}_G + \Delta\mathbf{g}_{j,OG} \text{ , with}$$
$$\Delta\mathbf{g}_{j,OG} = \begin{pmatrix} \left(1 + K_{peG}\right)^t \left(1 + K_{leG} t\right) r_j^0 \cos\theta_j^t \\ \left(1 + K_{peG}\right)^t \left(1 + K_{leG} t\right) r_j^0 \sin\theta_j^t \end{pmatrix}, \quad (28)$$

where $\mathbf{p}_{OG}$ is the center position of the image pattern at $t = 0$, $K_{peG} = 0$, $K_{leG} = K_{eG}$ if the isometric deformation is linear or $K_{peG} = K_{eG}$, $K_{leG} = 0$ if it is geometrical, $r_j^0 = \|\mathbf{g}_j^{t_0} - \mathbf{p}_{OG}\|$ is the distance between the goal position $\mathbf{g}_j^t$ and the center of the pattern $\mathbf{p}_{OG}$ at $t = 0$, and $\theta_j^t = \theta_j^0 + \omega_G t$ is the angle between the goal position $\mathbf{g}_j^t$ and the center of the pattern $\mathbf{p}_{OG}$ at time t. Both rotation and deformation are accounted for in the term $\Delta\mathbf{g}_{j,OG}$. Note

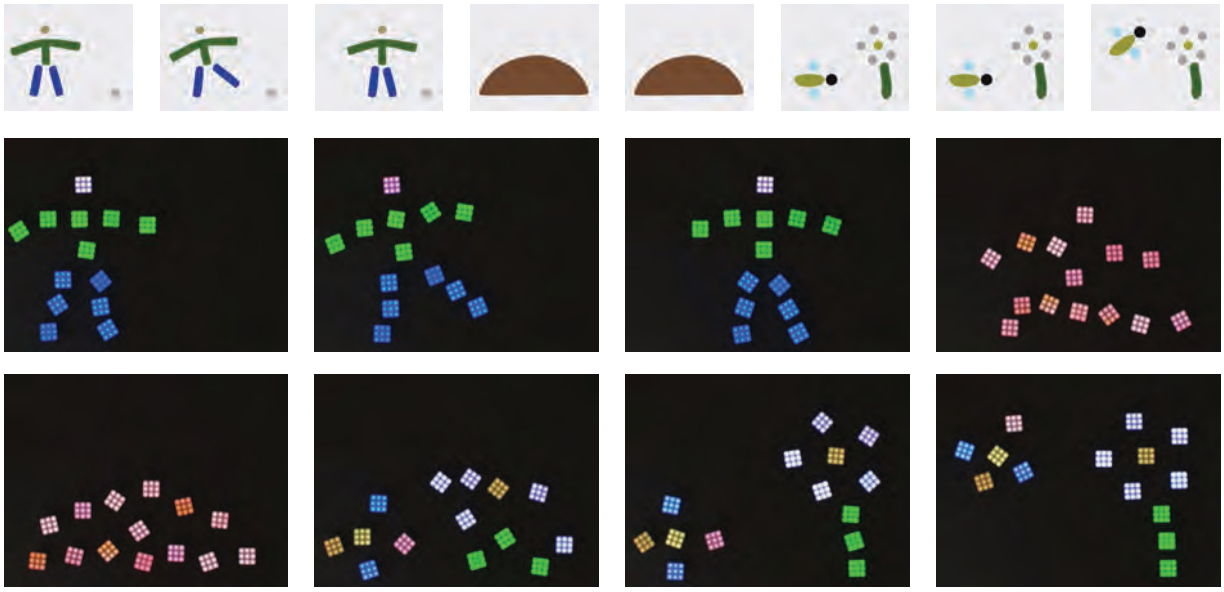

Fig. 16. Animation of a walking human, transitioning to a heap, transitioning to a flower circled by a bee. Artist's input images are shown in the top row, and corresponding images in the robot display are shown in the bottom two rows. The transitions from the human to the heap, and from the heap to the flower and the bee, are not prescribed but are determined by the system algorithms. The bottom two rows are real images of the robot display. The full video is available as Extension 1.

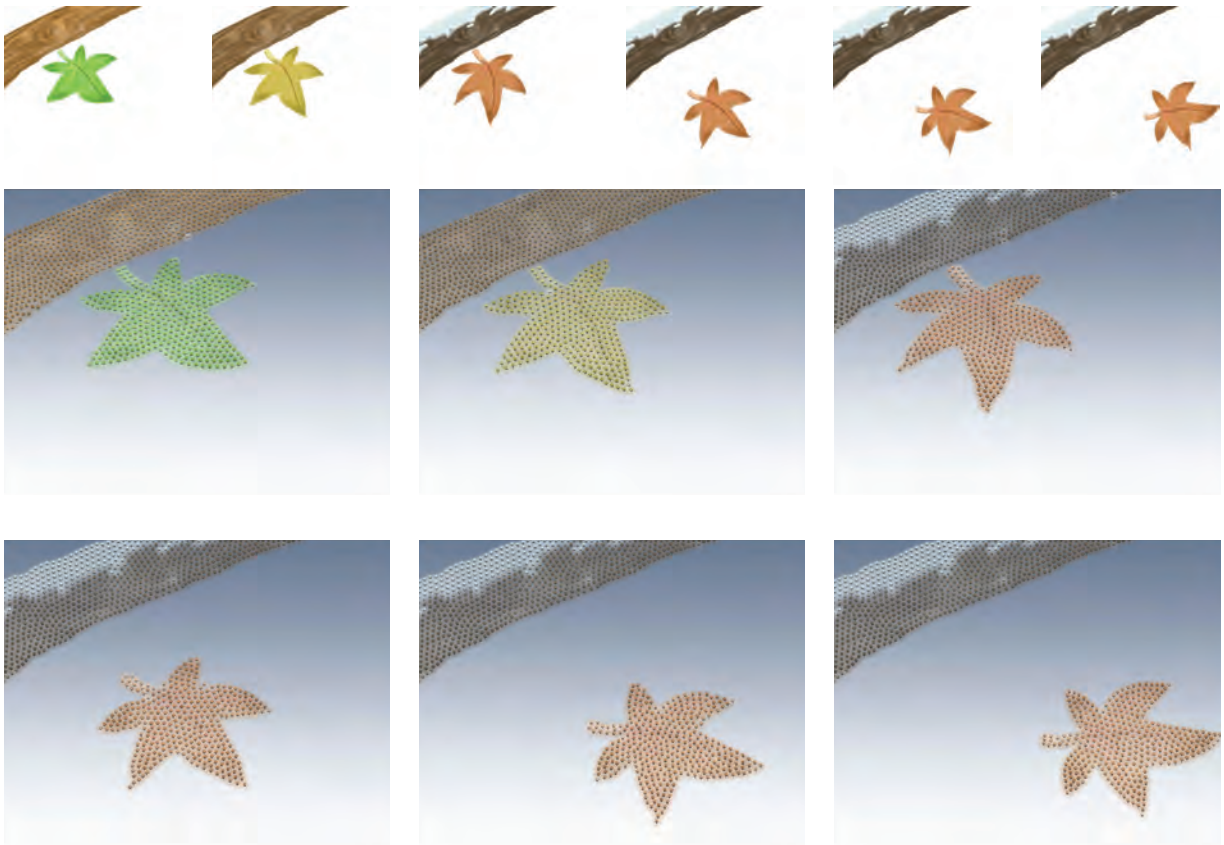

Fig. 17. Animation of a leaf that grows in the spring, withers in the fall, and falls from the branch in winter, using 1,000 simulated robot pixels. Artist's input images are shown in the top row, and corresponding images in the robot display are shown in the bottom two rows. The full video is available as Extension 1.

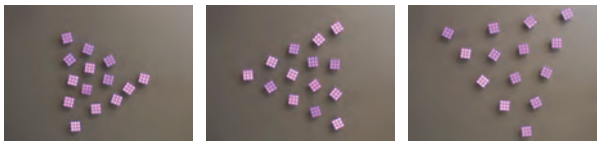

Fig. 18. Dynamic image display, with 14 e-puck robots show a continuously rotating and enlarging triangle. The full video is available as Extension 1.

Fig. 19. Dynamic obstacles. The green robot pixels are part of the normal display and are showing a static disk pattern. The red and blue robots are uncontrolled objects moving through the workspace. The green pixels respond in order to avoid collision with the dynamic obstacles. The full video is available as Extension 1.

Fig. 20. Perturbation recovery, with 14 e-puck robots form a rotating disk pattern. Two of the robots are picked up and placed outside the disk. The formation recovers and the two robots are incorporated smoothly back into the disk. The full video is available as Extension 1.

that the extension to the case of piecewise constant velocities and deformations is straightforward. In the case of constant velocity and deformation, the instant velocities $v^t_{g_j}$ of each goal position $\mathbf{g}^t_j$ can be computed by differentiation of Equation (28).

Figure 18 shows an example with a rotating and enlarging triangle. The robots, starting from a random configuration, converge to the given shape and follow it closely without breaking the contour. This is in contrast to the challenges observed for the animation display in Section 6. The speed of the dynamic image must lie within the robots' maximum speed or the robots are reassigned, resulting in undesired internal movements of the displayed image, or loss of shape.

7.4.2. Dynamic obstacles The avoidance of collisions with dynamic obstacles, mobile objects not under the control of the system, is handled at the level of local collision avoidance. Velocities of the dynamic obstacles are determined, and each robot pixel takes responsibility to avoid a collision. In Figure 19, the green robots are part of the normal display and are showing a static disk pattern. The red and blue robots represent dynamic obstacles not under control of the system: the red robot follows a given trajectory while the blue robot pursues the red robot. This experiment demonstrates correctly functioning collision avoidance with both controlled robot pixels and uncontrolled objects operating in the same workspace.

7.4.3. Perturbation recovery The system allows for fast and smooth recovery from strong perturbations, increasing the robustness against detection failures and allowing interaction, where several robots are moved to different locations by a human. In Figure 20 three images of an experiment in which two robots are moved to arbitrary positions by a human are showed. The system smoothly recovers the target configuration via a goal reassignment.

7.5. Large-scale display

This section describes a presentation of the system at Scientifica 2011, an exhibition held at ETH Zurich with the goal of bringing science to the public. The theme was 'The History of the Universe as told by The Robots', and 50 Elisa robots ran for 5 minutes every 15 minutes over a period of 3 days.

7.5.1. Experimental setup The experimental setup (camera, radio, and computer) is as described in Section 7.1. To ensure 10 Hz control in a standard four-core computer, the localization, communication, and control run in three different threads. The collision avoidance algorithm is parallelized, using one thread for each of the robots.

The Elisa robot, the small differentially driven robot shown in Figure 11, is a novel design which was specifically created to be a robot pixel. It has a diameter of 0.05 m and can perform fast motions of up to 0.5 m s^{-1} (although in our current setup it is limited to 0.25 m s^{-1}). It has a light diffuser over a single RGB LED. To ensure system scalability, the Elisa robot is a minimal platform that allows for low cost and autonomy. It consumes between 15 mW (sleep mode) and 1 W (motor and LEDs on) to provide a battery lifetime of 6 and 2 hours, respectively. The robot has three infrared LEDs, one on the front and two at the rear. All three infrared LEDs are used when initializing a robot's location or for recovery of a robot lost by the tracker. But only the frontal infrared LED is turned on during normal tracking and used to determine orientation. This approach is suitable for a large number of robots. The robot is able to perform automatic docking in recharging docks around the deployment area, reflecting the fact that manual recharging of batteries rapidly becomes intractable with increasing number of robots.

7.5.2. Animation display The robots were deployed on a black circular tabletop of 2 m diameter, as shown in Figure 1 and in the full length video in Extension 2. Figure 21 shows selected images from the animation: the big bang, the Solar system with Sun, Earth and Moon, a zoom onto planet Earth, the sea and the appearance of a microbe and a fish, a

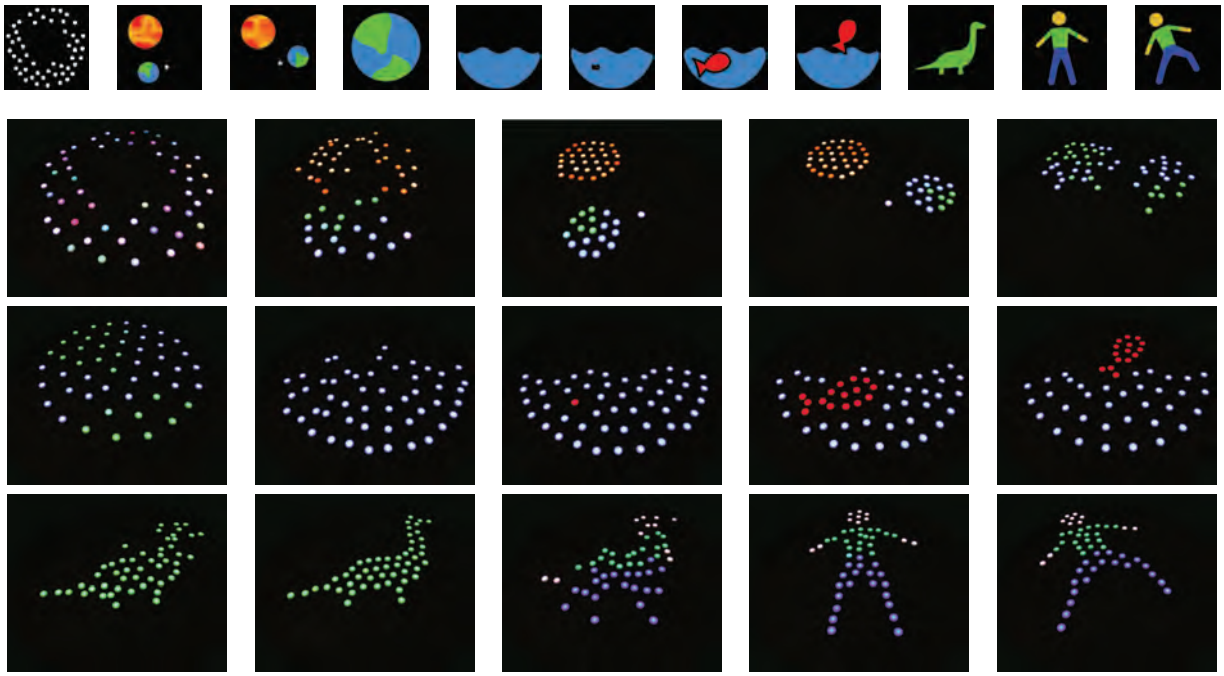

Fig. 21. Fifty Elisa robots display the history of the universe. The first images show the big bang, and then the Solar system with the Earth orbiting around the Sun and Moon orbiting around the Earth. A zoom in on the Earth is followed by the evolution of life. A microbe and a fish emerge in water, a dinosaur stretches its neck, and a human being arrives. The top row shows some of the input images with the corresponding robot images in the lower three rows, which are real images of the robot display. The full video is available as Extension 2.

dinosaur, and a human. This final experiment demonstrates several properties of the system: robustness of the system with the robots working on a repeated schedule over several days in a public environment, the ability to display diverse scenes, and scalability of our approach to a large swarm of 50 real robots.

8. Conclusion

This article has described a novel display in which each pixel is a mobile robot of controllable color, with a fully automatic method for showing representational images and animations. We presented a complete system that takes an image or animation as input and produces a multi-robot display as output. Experimental results have been shown for two real swarms of 14 and 50 robots and for simulations with 1,000 robotic pixels.

We showed that the system addresses several practical needs for a robot display formed using mobile robots. First, it adapts to the available number of robots: the system produces an optimal representation of a target image with a given size of the swarm, and it increases representational detail as more robots are added. Second, the algorithms scale well to large swarms (the collision-avoidance layer can be distributed). Third, the system can handle physical rearrangements of robots at run-time, supporting user interaction. Fourth, the method applies to a wide range of robot kinematics including holonomic, differential-drive and car-like kinematics, so it is suitable for a variety of applications and robots.

Each step in the method potentially supports distributed algorithms, although this would involve significant research challenges in on-board localization, communication, and computational power in small size robots. For future work, we are investigating 3D display with aerial vehicles (for which we believe the algorithms presented in this paper readily extend) and the problem of representing real images using a limited budget of robot pixels.

Funding

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Acknowledgments

The authors would like to thank G Caprari and GCtronic for the two robot swarms, S Hauri and S Haag for their work on the Elisa swarm, M Nitti for the input artwork, and Fortissimo Productions for the soundtrack of ‘The History of the Universe as told by The Robots’. The authors thank the anonymous reviewers for their helpful comments which greatly improved the quality of this article.

References

Alonso-Mora J, Breitenmoser A, Beardsley P and Siegwart R (2012) Reciprocal collision avoidance for multiple car-like robots. In *Proceedings IEEE International Conference on Robotics and Automation*.

Alonso-Mora J, Breitenmoser A, Rufli M, Beardsley P and Siegwart R (2010) Optimal reciprocal collision avoidance for multiple non-holonomic robots. In *Proceedings of the International Symposium on Distributed Autonomous Robotics Systems*.

Alonso-Mora J, Breitenmoser A, Rufli M, Haag S, Caprari G, Siegwart R, et al. (2011a) Displayswarm: A robot swarm displaying images. In *Symposium: Robot Demonstrations at International Conference on Intelligent Robots and Systems*.

Alonso-Mora J, Breitenmoser A, Rufli M, Siegwart R and Beardsley P (2011b) Multi-robot system for artistic pattern formation. In *Proceedings IEEE International Conference on Robotics and Automation*.

Bahceci E, Soysal O and Sahin E (2003) *A Review: Pattern Formation and Adaptation in Multi-robot Systems*. Technical Report, CMU.

Balch T and Hybinette M (2000) Social potentials for scalable multi-robot formations. In *Proceedings IEEE International Conference on Robotics and Automation*, vol. 1, pp. 73–80.

Belta C and Kumar V (2004) Abstraction and control for groups of robots. *IEEE Transactions on Robotics* 20: 865–875.

Bertsekas DP (1988) The auction algorithm: A distributed relaxation method for the assignment problem. *Annals of Operations Research* 14: 105–123.

Bertsekas DP and Castañon DA (1991) Parallel synchronous and asynchronous implementations of the auction algorithm. *Parallel Computing* 17: 707–732.

Breitenmoser A, Schwager M, Metzger J, Siegwart R and Rus D (2010) Voronoi coverage of non-convex environments with a group of networked robots. In *Proceedings IEEE International Conference on Robotics and Automation (ICRA)*.

Bullo F, Cortés J and Martínez S (2009) *Distributed Control of Robotic Networks*. Princeton, NJ: Princeton University Press.

Cianci CM, Nembrini J, Prorok A and Martinoli A (2008) Assembly of configurations in a networked robotic system: A case study on a reconfigurable interactive table lamp. In *IEEE Swarm Intelligence Symposium*.

Cortes J, Martinez S, Karatas T and Bullo F (2004) Coverage control for mobile sensing networks. *IEEE Trans Robotics and Automation* 20: 243–255.

Das A, Fierro R, Kumar V, Ostrowski J, Spletzer J and Taylor C (2002) A vision-based formation control framework. *IEEE Transactions on Robotics and Automation* 18: 813–825.

Desai J, Ostrowski J and Kumar V (1998) Controlling formations of multiple mobile robots. In *Proceedings IEEE International Conference on Robotics and Automation*, vol. 4, pp. 2864–2869.

Deussen O, Hiller S, van Overveld C and Strothotte T (2000) Floating points: A method for computing stipple drawings. *Computer Graphics Forum* 19: 40–51.

Du Q, Faber V and Gunzburger M (1999) Centroidal Voronoi tessellations: Applications and algorithms. *SIAM Review* 41: 637–676.

Ekanayake S and Pathirana P (2007) Geometric formations in swarm aggregation: An artificial formation force based approach. In *Third International Conference on Information and Automation for Sustainability, 2007 (ICIAFS 2007)*, pp. 82–87.

Fiorini P and Shiller Z (1998) Motion planning in dynamic environments using velocity obstacles. *The International Journal of Robotics Research* 17: 760–772.

Flyfire (2010) http://senseable.mit.edu/flyfire.

Gayle R, Moss W, Lin M and Manocha D (2009) Multi-robot coordination using generalized social potential fields. In *Proceedings IEEE International Conference on Robotics and Automation*.

Gazi V (2005) Swarm aggregations using artificial potentials and sliding-mode control. *IEEE Transactions on Robotics* 21: 1208–1214.

Hsieh MA, Kumar V and Chaimowicz L (2008) Decentralized controllers for shape generation with robotic swarms. *Robotica* 26: 691–701.

Hsu HC-H and Liu A (2005) Applying a taxonomy of formation control in developing a robotic system. In *17th IEEE International Conference on Tools with Artificial Intelligence, 2005 (ICTAI 05)*, pp. 3–10. ISSN 1082-3409.

Ikemoto Y, Hasegawa Y, Fukuda T and Matsuda K (2005) Gradual spatial pattern formation of homogeneous robot group. *Inf. Sci. Inf. Comput. Sci.* 171: 431–445.

Inaba M, Katoh N and Imai H (1994) Applications of weighted Voronoi diagrams and randomization to variance-based clustering (extended abstract). In *Symposium on Computational Geometry'94*, pp. 332–339.

Jacobsson M, Fernaeus Y and Holmquist LE (2008) Glowbots: Designing and implementing engaging human–robot interaction. *Journal of Physical Agents* 2: 51–59.

Ji M and Egerstedt M (2007) Distributed coordination control of multiagent systems while preserving connectedness. *IEEE Transactions on Robotics* 23: 693–703.

Kuhn HW (1955) The Hungarian method for the assignment problem. *Naval Research Logistics* 2: 83–97.

Lagae A and Dutré P (2006) *A Comparison of Methods for Generating Poisson Disk Distributions*. Report CW 459, Departement Computerwetenschappen, Katholieke Universiteit Leuven, Heverlee, Belgium.

Latombe J-C (1991) *Robot Motion Planning*. Boston, MA: Kluver.

LaValle SM (2006) *Planning Algorithms*. Cambridge: Cambridge University Press.

Lawton J, Beard R and Young B (2003) A decentralized approach to formation maneuvers. *IEEE Transactions on Robotics and Automation* 19: 933–941.

Lloyd S (1982) Least squares quantization in PCM. *IEEE Transactions on Information Theory* 28: 129–137.

McLurkin J and Smith J (2004) Distributed algorithms for dispersion in indoor environments using a swarm of autonomous mobile robots. In *International Symposium on Distributed Autonomous Robotic Systems*.

Michael N and Kumar V (2008) Controlling shapes of ensembles of robots of finite size with nonholonomic constraints. In *Proceedings of Robotics: Science and Systems IV*, Zurich, Switzerland.

Mondada F, Bonani M, Raemy X, Pugh J, Cianci C, Klaptocz A, et al. (2009) The e-puck, a robot designed for education in engineering. In *Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions*, vol. 1, pp. 59–65.

Okabe A and Suzuki A (1997) Locational optimization problems solved through voronoi diagrams. *European Journal of Operational Research* 98: 445–456.

Pimenta L, Kumar V, Mesquita R and Pereira G (2008) Sensing and coverage for a network of heterogeneous robots. In *Proceedings of the 47th IEEE Conference on Decision and Control*, pp. 1–8.

Ravichandran R, Gordon G and Goldstein SC (2007) A scalable distributed algorithm for shape transformation in multi-robot systems. In *Proceedings IEEE International Conference on Intelligent Robots and Systems*.

Rounds S and Chen Y (2009) Dynamic formation control using networked mobile sensors and centroidal voronoi tessellations. *ASME Conference Proceedings* 490: 109–118.

Rubenstein M and Shen W (2010) Automatic scalable size selection for the shape of a distributed robotic collective. In *Proceedings IEEE International Conference on Intelligent Robots and Systems*.

Secord A (2002) Weighted voronoi stippling. In *Proceedings of NPAR*. New York: ACM Press, pp. 37–43.

Szeliski R (2011) *Computer Vision Algorithms and Applications* (*Texts in Computer Science*). Springer.

Takahashi S, Yoshida K, Kwon T, Lee KH, Lee J and Shin SY (2009) Spectral-based group formation control. In *Computer Graphics Forum*.

Tsao J (2009) *Curious Displays*. Master's thesis, Art Center College of Design, Pasadena, CA.

van den Berg J, Guy SJ, Lin M and Manocha D (2009) Reciprocal *n*-body collision avoidance. In *International Symposium on Robotics Research (ISRR)*.

Varghese B and McKee G (2009) Towards a unifying framework for pattern transformation in swarm systems. In *Proceedings of the AIP Conference* 1107: 65–70.

Yang P, Freeman RA, Gordon G, Lynch K, Srinivasa S and Sukthankar R (2008) Decentralized estimation and control of graph connectivity in mobile sensor networks. In *American Control Conference*.

Yun S, Hjelle D, Schweikardt E, Lipson H and Rus D (2009) Planning the reconfiguration of grounded truss structures with truss climbing robots that carry truss elements. In *Proceedings IEEE International Conference on Robotics and Automation*.

Zavlanos MM, Spesivtsev L and Pappas GJ (2008) A distributed auction algorithm for the assignment problem. In *Proceedings IEEE International Conference on Decision and Control*.

Appendix: Index to Multimedia Extensions

The multimedia extension page is found at http://www.ijrr.org

Table of Multimedia Extensions

Extension	Type	Description
1	Video	Experiments from Section 7 with image display, animation display, and the extensions to dynamic image display, dynamic obstacles and perturbation recovery
2	Video	Experiment from Section 7.5 where 50 robots display the 'History of the Universe'