Autonomous Robots
https://doi.org/10.1007/s10514-021-09975-8

Herding by caging: a formation-based motion planning framework for guiding mobile agents

Haoran Song[1] · Anastasiia Varava[2] · Oleksandr Kravchenko[2] · Danica Kragic[2] · Michael Yu Wang[1] · Florian T. Pokorny[2] · Kaiyu Hang[3]

Received: 28 February 2020 / Accepted: 9 February 2021

Abstract
We propose a solution to the problem of *herding by caging*: given a set of mobile robots (called herders) and a group of moving agents (called sheep), we guide the sheep to a target location without letting them escape from the herders along the way. We model the interaction between the herders and the sheep by defining virtual "repulsive forces" pushing the sheep away from the herders. This enables the herders to partially control the motion of the sheep. We formalize this behavior topologically by applying the notion of *caging*, a concept used in robotic manipulation. We demonstrate that our approach is provably correct in the sense that the sheep cannot escape from the robots under our assumed motion model. We propose an RRT-based path planning algorithm for herding by caging, demonstrate its probabilistic completeness, and evaluate it in simulations as well as on a group of real mobile robots.

Keywords Topological representation and abstraction of configuration spaces · Computational geometry · Path planning for multiple mobile robots or agents · Motion and path planning

1 Introduction

In this paper, we consider a problem of planning motion for a team of robots to guide a group of mobile agents to a specified goal region. The agents are referred to as *sheep*, and the robots are called *herding robots*.

Despite the choice of terminology, the possible applications of our problem are not limited to herding animals (Lien et al. 2005; Strömbom et al. 2014; Vaughan et al. 2000). For instance, one can use several mobile robots working cooperatively to evacuate people during emergency situations, (Garrell et al. 2009). Moreover, one could use the robot team to either secure a team of people or to isolate a group of dangerous mobile objects, e.g., drones, from humans. A similar approach can also be applied to a team of robots to collect oil leaked on water (Bhattacharya et al. 2015) or to keep animals away from the runways in airports etc. (Lien et al. 2005).

We assume that the motion of the sheep is not directly controllable, but is restricted by the driving forces imposed by the herding robots. These forces are modelled as a distance-based potential field, and we assume that the sheep move

Haoran Song and Anastasiia Varava have contributed equally in this work.

This is one of the several papers published in Autonomous Robots comprising the Special Issue on Topological Methods in Robotics.

✉ Haoran Song
hsongad@ust.hk

Anastasiia Varava
varava@kth.se

Oleksandr Kravchenko
okr@kth.se

Danica Kragic
dani@kth.se

Michael Yu Wang
mywang@ust.hk

Florian T. Pokorny
fpokorny@kth.se

Kaiyu Hang
kaiyu.hang@yale.edu

[1] Robotics Institute, Hong Kong University of Science and Technology, Hong Kong, China

[2] Robotics, Perception and Learning Lab, KTH Royal Institute of Technology, Stockholm, Sweden

[3] Department of Mechanical Engineering and Material Science, Yale University, New Haven, USA

Published online: 10 March 2021

such that they want to never decrease the distance to the herding robots. In our setting, the herding team forces the sheep to move from an initial state to a specified goal region, while ensuring that the sheep do not escape from the herders.

The motion process consists of two alternating phases. In the *repulsion* phase, the herding robots remain static, and the sheep move away from the herders. A potential function strictly monotonically decreases as the distance to the closest robot increases. The motion of the sheep is restricted by this potential field, as they aim to never increase their potential during the repulsion phase. Therefore, we can control their motion by surrounding them with the robots, so that the robots form a closed region of high potential around the flock. Since the sheep never increase their potential, they cannot escape from the herders as long as they are surrounded by a region of high potential. In contrast, in the *herding* phase, when both the sheep and the robots move, the sheep can move in any direction with a bounded velocity. We do not make any assumptions about the direction of their motion. As such, our system is partially controlled: we affect the motion of the sheep only in the repulsion phase.

To provide a provable guarantee that that the sheep never escape from the herders, we adopt the concept of *caging* to formally describe the situation in which the motion of the sheep is restricted by the herders. We address the caging verification problem by computing homology groups of superlevel sets of the potential function to check that the sheep are located in a bounded region of low potential, and therefore cannot escape from the robots. We represent the superlevel sets as alpha shapes (Edelsbrunner and Harer 2010).

This paper extends our previous work (Varava et al. 2017) and presents contributions on both the theoretical and the experimental sides. We address the problem of initial cage acquisition and provide algorithms for two different scenarios: (i) when the number of herding robots is predefined and the goal is to optimize the time needed to form a cage, and (ii) when the goal is to minimize the number of herding robots while prioritizing caging formations that can be reached quickly. Moreover, we perform extensive experiments with a group of mobile robots. Experimental scenarios include cage acquisition, motion in a cluttered environment and with a narrow passage, and team reformation in case one or more of the herding robots breaks.

2 Related work

Motion planning for a team of mobile robots is an essential problem in many real world applications, such as the coverage control for mobile sensing networks (Cortes et al. 2004), behavior-based control for robot teams (Balch and Arkin 1998; Lee and Kim 2017), as well as communication-constrained motion planning for multi-robot systems (Pereira et al. 2003, 2004), etc. In research on robot formation control and motion planning, the problem formulations can be classified into three groups (Beard et al. 2001; Garrido et al. 2011): (1) A robot in a team is designated as the leader, and is first commanded to follow a predefined pose trajectory (position and orientation) to lead the team. The other robots in the team are moving by following the leader while having to satisfy a set of task-related geometric constraints (Tanner 2004; Elamvazhuthi et al. 2020); (2) With the concept of virtual structure, the robot team is modeled as a single structure, in which the motion of each robot is translated from the desired global structure (Egerstedt and Hu 2001a); and (3) The robot team is desired to provide a group behavior, and each single robot's motion is subject to a weighted average of several behaviors (Balch and Arkin 1998).

To the best of our knowledge, the research on formation control has been mainly focused on the control aspect of the formation maintenance and the motion planning for achieving the desired formations. The problem of how a team of mobile robots can interact with moving agents, such as a group of people or animals, and steer them to a goal region, has received surprisingly little attention in the literature. Schultz et al. (1996) addressed the problem using genetic algorithms and neural networks to model the local behaviors of flock control. In a work by Vaughan et al. (2000), a single robot communicates with a central vision system to choose controllers to drive the herd of ducks to the goal. Alternatively, certain controllable parts of the environment (gates or other movable obstacles) can act as agents, allowing to steer the non-controlled mobile robots by changing the geometric or topological properties of the scene (Durrant-Whyte et al. 2012; Fine and Shell 2013).

When modeling the passive agents' behavior, several researchers have assumed that they are "repulsed" from the herders. Lien et al. (2005) provided the first work on shepherding behaviors with multiple shepherds and a large flock size. They studied how multiple shepherd agents can control another group of agents. They assumed that the passive agents react based on repulsive forces exerted by the shepherds and obstacles in the environment. In a study by Bacon and Olgac (2012), the authors used a sliding model controller to place herders around a single evader to drive it along a desired trajectory. The evader's motion was modeled through repulsion forces exerted by the herders within a certain sensing range.

Strömbom et al. (2014) proposed a self-propelled particle model of local attraction–repulsion type to model herding of a group of agents by one shepherd. In a work by Garrell et al. (2009), the authors worked on guiding people in the environment represented as a potential field, which enables the authors to guide people in urban areas. Artificial potential field-based controllers for herding have also been used by Tanner et al. (2007) and Gazi and Passino (2004). (Pier-

son and Schwager 2018) designed control strategies to steer noncooperative herds.

Apart from that, herding has also been formulated as a pursuit-evasion game (Lu 2010; Shedied 2002). In the works of Egerstedt and Hu (2001b) and Ferrari-Trecate et al. (2006), agents are being relocated by driving a formation of to a goal, based on an assumption that the sheep cooperate with the herders. The problem of herding cows using smart collars equipped with GPS and sound amplifiers was also considered by Butler et al. (2004).

The major difference between all the above mentioned works and ours is that we employ *caging* formalism used in robotic manipulation to *safely* (i.e., without letting them to escape at any moment of time) and *with provable guarantees* guide a group of mobile agents to the goal region. Caging is a way of restricting the mobility of an object without immobilizing it completely. To the best of our knowledge, this is the first application of caging in the context of steering moving agents. So far, the notion of caging has been mostly used in robotic grasping to partially immobilize an object under consideration (Makita and Wan 2017), or as a waypoint to a form-closure grasp (Rodriguez et al. 2012). Several recent works by Mahler et al. (2016a, b) extend the notion of caging to energy-bounded caging, where physical obstacles and energy fields (such as gravity) are utilized to restrict the mobility of an object. These works are somewhat related to ours as we deal with potential-based caging instead of bounding the mobility of the sheep by physical obstacles. However, the caging verification method proposed in our work is different from the works by Mahler et al. (2016a, b), where the authors consider a single static object and explicitly approximate its entire three-dimensional configuration space. Instead, we consider a set of moving agents, and make sure that they are always surrounded by a region with high potential values by working directly in the two-dimensional workspace.

Several works use the caging formalism to enable groups of mobile robots to move objects. In a work by Fink et al. (2008), a caging-based approach to multirobot manipulation has been proposed. The authors proposed a fully-controlled contact-based method for object transportation, while ours achieves escorting moving agents by the partially-controlled potential-based motion process. In their work, the object, which is a rigid shape, is passively moved by the robots, while in our work, a group of sheep is actively herded by the robots by the induced virtual repulsive forces. In a work by Bhattacharya et al. (2015), the authors use an approach to separate and manipulate sets of objects using cables. The main difference between these works and ours is that instead of moving static objects, we steer moving agents, and thus address the aspect of modeling and indirectly controling their motion.

3 Problem formulation

3.1 Potential-based caging

To address our problem, we first define the notion of a cage. According to the classical definition, an object is *caged* by a caging tool if it cannot escape arbitrarily far from the caging tool (i.e., a manipulator in robotic grasping, or a group of robots in our context). In our setting, a sheep is caged if it is surrounded by a region of potential higher than its own, so that it would have to temporarily increase its potential in order to escape from the robots.

In our work, we abstract both the sheep and the robots as points. The goal of this paper is to define and propose a solution to the *cage-steering* problem: starting from initial position, where the flock is surrounded by the herders, we move the latter in such a way that the repulsive forces induced by them push the flock to some predefined location.

Consider a workspace $\mathcal{W} \subset \mathbb{R}^2$. Let $\mathcal{S} = \{s_1, s_2, \ldots, s_m\}$ denote the set of sheep, and $\mathcal{R} = \{r_1, r_2, \ldots, r_n\}$ denote the set of robots controlling their motion.[1] We assume that the robots move in $\mathcal{W}$, and therefore the configuration space $\mathcal{C}$ of the team is a subset of $\mathcal{W}^n$. Let us denote the collision-free subset of $\mathcal{C}$ by $\mathcal{C}_{free}$.

Let $\mathrm{d}_c : \mathcal{W} \to \mathbb{R}_{\geq 0}$ denote, for a given configuration c, the distance from a point to closest robot from the set $\mathcal{R}$ in this configuration. Assume that the sheep tend to keep away from the robots. To formally describe their behaviour, we introduce a potential function $p_c : \mathcal{W} \to \mathbb{R}$.

Definition 1 A potential function $p_c : \mathcal{W} \to \mathbb{R}$ is a continuous function strictly monotonically decreasing with the distance from $x \in \mathcal{W}$ to the closest robot from $\mathcal{R}$, when the robots are at the configuration $c \in \mathcal{C}_{free}$.

Since the potential of any point $x \in \mathcal{W}$ is uniquely defined by the distance to the closest robot, it is convenient to use the following notation: $p_d(\mathrm{d}_c(x)) = p_c(x)$, where $p_d : \mathbb{R}_{\geq 0} \to \mathbb{R}$ is a distance-based potential function, and $p_c : \mathcal{W} \to \mathbb{R}$ is a point-based function defined for configuration c.

3.2 The motion model

Let us now model the interaction between the sheep and the robots. We consider two types of behavior: *repulsion* and *herding*. These two types represent two different phases of the steering process. During the repulsion phase, the robots do not move, while the sheep can move with respect to them. During the herding phase, both the robots and the sheep move. In practice, the motion process consists of a sequence

[1] The number of robots n does not depend on m, but, in our work, should not be smaller than 3 in the case of a 2D workspace. Robots are all the same and indistinguishable to the sheep.

of alternating repulsion and herding phases. We assume that the speed of the sheep never exceeds v_s. Let us first describe the repulsion phase.

3.2.1 Repulsion phase

Let $s(t)$ denote the trajectory of a sheep in $\mathcal{W}$ during some period of time $t \in [0, T]$. Assume that the robots stay at the same configuration $c \in \mathcal{C}_{free}$. Then the sheep never moves to points of higher potential:

$$\forall t_1 < t_2 \in [0, T] : \; p_c(s(t_1)) \geq p_c(s(t_2)).$$

Note that this assumption allows a more general class of motion than just following the gradient of the potential field, although the latter is a valid example. This also implies that the position of the robots does not uniquely define the motion of the flock, but rather partially restricts it.

3.2.2 Herding phase

During herding phase, the robots move between two configurations $c_0, c_1 \in \mathcal{C}_{free}$. Even though the sheep might want to keep away from the herders during this phase as well, this might be infeasible due to several reasons. First, they may not be able to accurately predict the motion of the robots, and therefore might accidentally move closer to them. Second, the computation of a trajectory that allows the sheep to keep as far away as possible from the herders requires a good sense of orientation and fast reaction (in the applications where by "sheep" we mean people or animals), and a certain computational capacity (when the passive agents are also robots). Moreover, their perception of the herding robots' positions might not be precise due to noise. To keep our setting as generic as possible, we do not make any assumptions on the direction of the sheep during this phase, and only assume that the speed of the sheep is bounded.

Note that while the potential value of a sheep does not increase during the repulsive phase, it might increase during the herding phase, see Fig. 1. This happens because during the herding phase the robots do not control the motion of the sheep, which makes the system only *indirectly controlled*. For this reason, the herding phase is limited in time in such a way that we can guarantee that during it the sheep will not escape from the robots, which is possible as the sheep move at a constant speed. In contrast, when the robots do not move (i.e., during the repulsion phase) the sheep move in a way that never increases their potential. Therefore, if the sheep are properly surrounded by a region of high potential, they never escape from the robots, and hence we do not have to limit the duration of this phase.

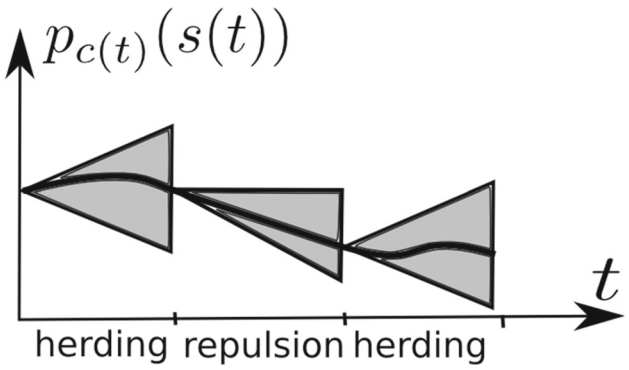


Fig. 1 Three alternating motion phases are depicted in this figure. The black curve reflects the actual dynamics of the potential value of a single sheep $s-p_{c(t)}(s)$, while the grey triangles correspond to the regions of possible values. The angle of the grey cones is determined by the sheep's maximal speed v_s

4 Mathematical background

In this section, we explain the concepts from computational topology we use later in the paper.

4.1 Holes, voids and homology groups

Algebraic topology aims to classify topological spaces up to continuous deformations. One of its main tools is the computation of homology groups. Intuitively, the elements of a *homology group* $H_i(\mathcal{X})$ of the space $\mathcal{X}$ represent the number of *voids* in $\mathcal{X}$. For instance, in a 2D space the elements of $H_1(\mathcal{X})$ correspond to the holes, while in a 3D space $H_2(\mathcal{X})$ represents its voids.

In this paper, we are interested in computing homology groups in two cases. When we consider a two-dimensional workspace $\mathcal{W}$, we construct its subset $\mathcal{X}$ in such a way that the potential value at any of its points is higher then in its complement $\mathcal{W}-\mathcal{X}$. This way, the sheep are caged when they are located in bounded subsets of lower potential, which can be seen as *holes* in $\mathcal{X}$. We compute the first homology group of $\mathcal{X}$ with coefficients in $\mathbb{Z}_2$ to find the holes. The elements of $H_1(\mathcal{X})$ are closed curves going around the holes, i.e., the curves that cannot be continuously deformed into a point in $\mathcal{X}$. Each hole corresponds to a homology class. Since these curves have high potential values, they bound the mobility of the sheep located in the holes, see Fig. 2.

In the case of a three-dimensional workspace $\mathcal{W}$, the bounded subsets of low potential correspond to the voids in the high potential subset $\mathcal{X}$ of $\mathcal{W}$, which are represented by its second homology group $H_2(\mathcal{X})$. Similarly to the two-dimensional case, the elements of $H_2(\mathcal{X})$ are subsets of $\mathcal{X}$, separating the voids from the remaining part of $\mathcal{W} - \mathcal{X}$.

4.2 Simplicial complexes

For computational reasons, it is convenient to work with discrete versions of spaces, which can be achieved by representing them as *simplicial complexes*.

A geometric $k-$simplex $\sigma = [v_0, \ldots, v_k]$ in $\mathbb{R}^d$ is a convex hull of $k+1$ ordered affinely independent elements $v_0, \ldots, v_k \in \mathbb{R}^d$. A convex hull of a subset of $\{v_0, \ldots, v_k\}$ is called a face τ of the simplex σ, which is denoted by $\tau \leq \sigma$. A finite simplicial complex $\mathcal{K}$ is a non-empty set of simplices such that:

- if $\tau \leq \sigma$, then $\tau \in \mathcal{K}$,
- if $\sigma, \sigma' \in \mathcal{K}$, then $\sigma \cap \sigma' = \emptyset$ or $\sigma \cap \sigma' \in \mathcal{K}$

In 2D, a simplicial complex $\mathcal{K}$ is a set of vertices, line segments and triangles, whose intersections are either empty or belong to $\mathcal{K}$. In 3D, a simplicial complex consists of vertices, segments, triangles and tetrahedra with the same property.

4.3 Topology of a union of balls: alpha complexes

In the next section, we deal with subsets of $\mathcal{W}$ represented as unions of closed balls of a fixed radius. In particular, we will be interested in the homology groups of these sets. Let $X = \{x_1, \ldots, x_n\}$ be a finite set of points in $\mathbb{R}^d$, and let $R > 0$ be a real number. Consider a union of closed balls with centers at points from X and radii R.

From the computational point of view, it is not easy to compute homology groups of $\bigcup_{i=1}^n B_R(x_i)$ directly. Fortunately, this is not necessary, as we can work with its discrete version—the *alpha complex*.

An alpha complex $A(R)$ corresponding to the union of balls $\bigcup_{i=1}^n B_R(x_i)$ is a simplicial complex with vertices $\{x_1, \ldots, x_n\}$ which lies strictly inside $\bigcup_{i=1}^n B_R(x_i)$, and is homotopy equivalent to the latter (Edelsbrunner and Harer 2010).

Given a set of points $X = \{x_1, \ldots, x_n\}$, we can continuously increase the radius and get a nested family of unions of balls. Correspondingly, we get a nested family of alpha complexes, $X = A(R_0) \subset A(R_1) \subset \cdots \subset D(X)$, where $D(X)$ is the simplicial complex corresponding to Delaunay triangulation of X that includes faces. Any alpha complex is a subcomplex of Delaunay triangulation of X, and since the latter is finite, the family of nested subcomplexes is also finite. In our work, we use this fact for cage verification.

5 Potential-based caging

We want to guarantee that the sheep are caged during repulsion phases, and that the herding phases are limited in time in such a way that the sheep do not escape from the robots. In this section we provide the necessary definition and formalize potential-based caging. Then, we derive sufficient conditions for safe moves between two caging configuration—i.e., conditions for the robots' motion during the herding phase. Later, we provide the algorithm for motion planning. From now on, we assume that $\mathcal{W}$ is two-dimensional. We discuss the possible generalization to the three-dimensional case later in this section.

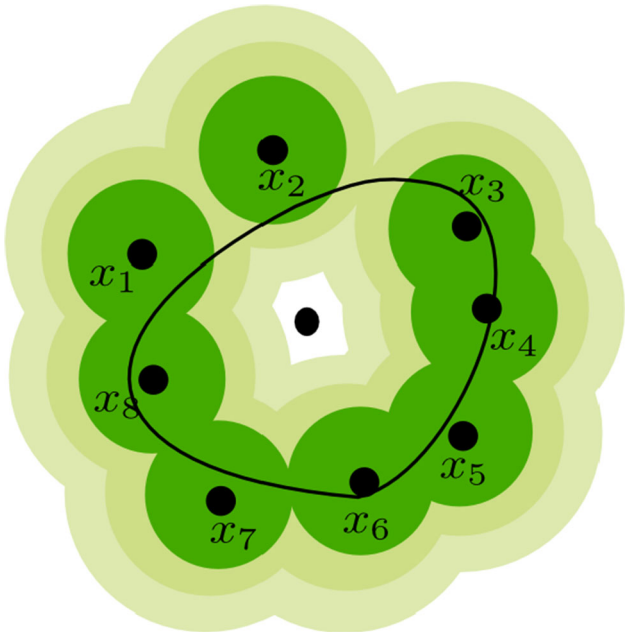


Fig. 2 The black curve is a high potential fence induced by the robots $\{x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8\}$. Different shades of green depict different superlevels of the potential function. The choice of the set O is not unique: e.g., any open set from the white and light green regions from the interior of the curve satisfies the required conditions (Color figure online)

Definition 2 Let $\mathcal{R}_c$ denote the set of robots in configuration c. A *high potential fence* $HPF(\mathcal{R}_c)$ induced by robots $\mathcal{R}$ in configuration c is a closed curve in $\mathcal{W}$, such that there exists a non-empty open set O in its interior,[2] the supremum of the potential value $\sup_{x \in O}(p_c(x))$ in which is strictly lower than the potential of the fence, defined as $p_c(HPF(\mathcal{R}_c)) = \min_{z \in HPF(\mathcal{R}_c)} p_c(z)$.

Figure 2 illustrates the concept introduced above. Note that in some situations we do not need to use all the robots from $\mathcal{R}$ to form a high potential fence. Note also that some configurations can induce several high potential fences.

Definition 3 A configuration $c \in \mathcal{C}$ is a *caging configuration* if there exists a high potential fence induced by $\mathcal{R}_c$.

[2] By interior of a closed curve in $\mathbb{R}^2$ we mean the union of bounded connected components in its complement. Note that the term "connected components" is used in the topological sense, and a closed curve can potentially have self-intersections.

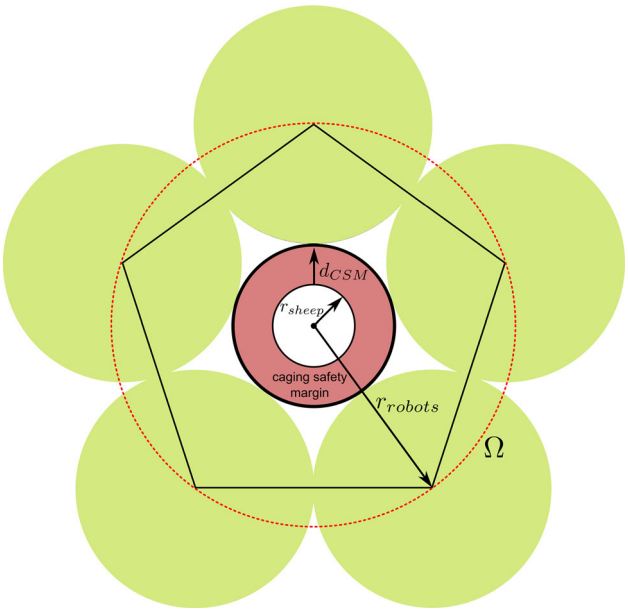


Fig. 3 This figure depicts a caging chain formed by 5 robots. r_{sheep} is the radius of the circle which is guaranteed to contain the sheep at the current moment of time. Green circles represent the area of potential, greater or equal to the potential of the caging chain. The red region corresponds to the caging safety margin. r_{robots} is the radius of the caging circle Ω (Color figure online)

Definition 4 A sheep *s is caged* by the robots if it is situated in the interior of some high potential fence, and its potential is strictly lower than the potential of the fence.

6 Cage acquisition

6.1 Smallest cage acquisition

We now describe the algorithms for initial cage acquisition. Here, we assume that we have the initial coordinates of herding robots and sheep, and the sheep can move in arbitrary directions with a constant speed v_s. We want to minimize the size of the resulting cage. The maximum[3] speed of the herding robots is denoted by v_r. Finally, the *cage safety margin size* d_{CSM} is the minimal required distance between the sheep and the point where the potential of a cage is the lowest, see Fig. 3.

Let us first introduce the necessary notation, see Fig. 3. Consider the smallest circle containing the initial locations of the sheep, and let r^0_{sheep} and c_{sheep} be the radius and the center of that circle at time moment 0. In other words, this circle is guaranteed to contain all sheep. Since in our algorithm we assume that sheep can move in arbitrary directions, the radius of the circle that is guaranteed to contain all sheep grows to r^t_{sheep} at time t. Our task is to make sure that it is caged by the herding robots by the end of the cage acquisition. For this, we construct a regular n-sided polygon and put herding robots at its vertices. This polygon is constructed as a regular polygon inscribed in a *caging circle* Ω—a circle of radius r_{robots} and center c_{sheep}. The value r_{robots} can be computed given r^t_{sheep} and caging safety margin size d_{CSM}:

$$r_{robots} - r^t_{sheep} - d_{CSM} = r_{robots} \cdot \sin(\pi/n) \quad (1)$$

This way, we ensure that the sheep never escape: between time steps 0 and t they are guaranteed to be within the circle of radius r^t_{sheep}, and after that this circle is located within a cage. The Algorithm 1 proceeds as follows: first, we select the estimated time t needed for acquiring a cage. We do this

[3] We assume that the robots do not move slower than at their maximum speed until they reach their final positions, so we assume that it is fixed and equal to v_r.

Algorithm 1: Cage acquisition (minimal cage size)

input : robots coordinates $(x_1, \ldots, x_n)$,
sheep coordinates $(y_1, \ldots, y_m)$,
cage safety margin d_{CSM},
sheep speed v_s,
herders speed v_r,
bisection search convergence threshold ε
output: robot trajectories $\{\gamma_1, \ldots, \gamma_n\}$

$r^{min}_{sheep} = 0$
r^{max}_{sheep} = field-size
(c_{sheep}, r^0_{sheep}) = CircumscribedCircle$((y_1, \ldots, y_m))$
while $r^{max}_{sheep} - r^{min}_{sheep} > \varepsilon$ **do**
  $r_{sheep} = (r^{max}_{sheep} + r^{min}_{sheep})/2$
  r_{robots} = CagingCircle(r_{sheep}, d_{CSM}, n)
  $t_{sheep} = (r_{sheep} - r^0_{sheep})/v_s$
  $t^{opt}_{rob} = t_{sheep}$
  $orient_{cur} = 0$
  while $orient_{cur} < 2\pi$ **do**
    for $(vert_j, rob_i), i, j \in \{1, \ldots, n\}$ **do**
      $\mathbf{T}_{v_j, r_i}$ = TimeForRobToVert(v_j, r_i)
    end
    Vertex = VertRobAssignment(**T**)
    $t^{max}_{rob} = \max_{rob \in \{1,\ldots,n\}}$ TimeToVer$(rob,$ **Vertex**$_{rob})$
    if $t^{max}_{rob} < t^{opt}_{rob}$ **then**
      $t^{opt}_{rob} = t^{max}_{rob}$
      $orient_{opt} = orient_{cur}$
    end
    $orient_{cur} = orient_{cur} + orient_{step}$
  end
  if $t^{opt}_{rob} < t_{sheep}$ **then**
    $r^{max}_{sheep} = r_{sheep}$
  else
    $r^{min}_{sheep} = r_{sheep}$
  end
end
for $rob \in \{1, \ldots, n\}$ **do**
  γ_{rob} = ComputeRobotTrajectories(**Vertex**$_{rob}$)
end
return $\{\gamma_1, \ldots, \gamma_n\}$

by performing bisection search on r^t_{sheep} between values 0 and the diameter of the field. For each r^t_{sheep}, we compute the necessary r_{robots} [see Eq. (1)], and t_{sheep}—the latest time moment at which the sheep are still guaranteed to be within a circle of radius r^t_{sheep}. Now when we have the center c_{sheep} and the radius r_{robots} of the caging circle, we want to compute the best $n-$sided polygon inscribed into it. For this, we need to determine its orientation. We do so by checking a finite set of possible orientations uniformly distributed from 0 to 2π with a step $orient_{step}$. For each orientation, we then assign robots to vertices of the resulting polygon using the least total time t^{max}_{rob} needed to reach those positions, which is implemented by VertRobAssignment function in Algorithms 1 and 2. Each herding robot follows a straight line leading it to the vertex it is assigned to. If t^{max}_{rob} does not exceed t_{sheep}, then it is possible to construct a cage by the moment t_{sheep}.

6.2 Cage acquisition with the minimal number of herding robots

In this version of the cage acquisition Algorithm 2, we want to minimize the necessary number of herding robots. Similarly to the previous case, we first perform bisection search on the size of the cage r^t_{sheep}, and as before, we compute the time t_{sheep} and the cage circle radius r_{robots}. At each iteration of the bisection search, we consider possible number of herding robots from 3 to n. Given a number of herding robots, we check if is possible to construct a cage by the time moment t_{sheep}.

Algorithm 2: Cage acquisition (minimal number of herders)

input : robots coordinates $(x_1, \ldots, x_n)$,
sheep coordinates $(y_1, \ldots, y_m)$,
cage safety margin d_{CSM},
sheep speed v_s,
herders speed v_r,
bisection search convergence threshold ε
output: robot trajectories $\{\gamma_1, \ldots, \gamma_n\}$

$overall\text{-}best\text{-}rob\text{-}num = n + 1$
$r^{min}_{sheep} = 0$
$r^{max}_{sheep} =$ field-size
$(c_{sheep}, r^0_{sheep}) =$ CircumscribedCircle$((y_1, \ldots, y_m))$
while $r^{max}_{sheep} - r^{min}_{sheep} > \varepsilon$ **do**
  $best\text{-}rob\text{-}num = n + 2$
  $r_{sheep} = (r^{max}_{sheep} + r^{min}_{sheep})/2$
  for $robot\text{-}num \in \{3, \ldots, n\}$ **do**
    $r_{robots} =$ CagingCircle$(r_{sheep}, d_{CSM}, robot\text{-}num)$
    $t_{sheep} = (r_{sheep} - r^0_{sheep})/v_s$
    $t^{opt}_{rob} = t_{sheep}$
    $orient_{cur} = 0$
    while $orient_{cur} < 2\pi$ **do**
      for $(vert_j, rob_i), i, j \in \{1, \ldots, n\}$ **do**
        $\mathbf{T}_{v_j, r_i} =$ TimeForRobToVert(v_j, r_i)
      end
      Vertex = VertRobAssignment(**T**)
      $t^{max}_{rob} = \max_{rob \in \{1,\ldots,n\}}$ TimeToVer$(rob,$ **Vertex**$_{rob})$
      if $t^{max}_{rob} < t^{opt}_{rob}$ **then**
        $t^{opt}_{rob} = t^{max}_{rob}$
        $orient_{opt} = orient_{cur}$
        best-rob-num $=$ ***robot-num***
      end
      $orient_{cur} = orient_{cur} + orient_{step}$
    end
    if $best\text{-}rob\text{-}num < n + 1$ **then**
      break
    end
  end
  if $best\text{-}rob\text{-}num \geq overall\text{-}best\text{-}rob\text{-}num$ **then**
    $r^{min}_{sheep} = r_{sheep}$
  else
    $r^{max}_{sheep} = r_{sheep}$
    $overall\text{-}best\text{-}rob\text{-}num = best\text{-}rob\text{-}num$
  end
end
for $rob \in \{1, \ldots, overall\text{-}best\text{-}rob\text{-}num\}$ **do**
  $\gamma_{rob} =$ ComputeRobotTrajectories(**Vertex**$_{rob}$)
end
return $\{\gamma_1, \ldots, \gamma_n\}$

7 Cage verification

Suppose we have an initial static configuration of robots and a set of sheep, and we want to check whether they form a cage. For that, we need to check whether the caging condition holds, i.e., if the robots form a high potential fence such that the sheep are located in its interior.

From the computational point of view, it is convenient to have a discrete version of the notion of high potential fence. Namely, we would like to deal with points and segments instead of continuous curves. We therefore introduce the following definition.

Definition 5 An ordered set of robots $(x_{a_1}, \ldots, x_{a_k})$ that form a high potential fence around a sheep with coordinates y, together with segments connecting each pair of consecutive robots, and the first and the last ones, forms a polygonal chain g called *caging chain*.

A caging chain (see Fig. 4) is a special case of a high potential fence. The advantage of this notion is that a caging chain is a closed curve consisting of a finite number of segments, connecting pairs of robots. In other words, a caging chain g formed by k robots with coordinates $(x_{a_1}, x_{a_2}, x_{a_3}, \ldots, x_{a_k})$ can be viewed as a set of pairs $g = \{(x_{a_1}, x_{a_2}), (x_{a_2}, x_{a_3}), \ldots, (x_{a_k}, x_{a_1})\}$. When the robots move, the caging chain also moves and deforms; as long as the movement preserves the cage, we can keep track of the deformations of the initial caging chain. Assume that at some moment of time t the new coordinates of the corresponding robots are $(x^t_{a_1}, x^t_{a_2}, x^t_{a_3}, \ldots, x^t_{a_k})$, and the cage has been pre-

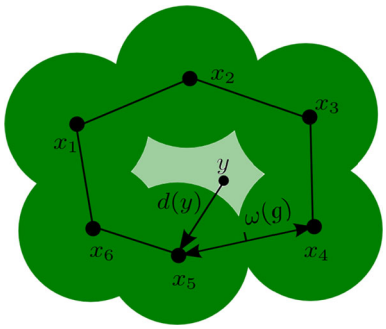


Fig. 4 This figure depicts a caging chain formed by 6 robots. $\omega(g)$ is the half of the length of the longest segment of the chain. Green circles depict the superlevel set of the potential function, containing the caging chain (Color figure online)

served along the way. By g_t we denote the deformation of the corresponding caging chain, which now can be considered as $g_t = \{(x^t_{a_1}, x^t_{a_2}), (x^t_{a_2}, x^t_{a_3}), \ldots, (x^t_{a_k}, x^t_{a_1})\}$.

The width of the caging chain is defined as a half of the length of its longest segment:
$\omega(g) = 1/2 \max_{(x_i,x_j)\in g}(\text{length}(x_i, x_j))$. This value is important, since it defines the lowest potential of the caging chain: $p_c(g) = p_d(\omega(g))$.

Consider a sheep $s \in \mathcal{S}$ with coordinates y, and assume that the robots in configuration $c \in \mathcal{C}$ have coordinates $(x_1, x_2, \ldots, x_n)$, respectively. To verify the caging condition, we first need to compute the high potential fences induced by c. For that, we study the topology of the superlevel sets of the potential function $p_c(y)$.

A superlevel set is a set of the form $L^+_q = \{x \in \mathcal{W} | p_c(x) \geq q\}$, where the value of the potential function is not lower than q. To check if a given configuration forms a cage, one can consider the topological properties of the sets defined above. Namely, if a sheep with coordinates $y \in \mathcal{W}$ and a potential value $p_c(y) < q$ for some positive real number q lies in the interior of some closed curve $\phi \subset L^+_q$, then ϕ is a high potential fence by definition. Therefore, the sheep is caged.

First of all, observe that any such curve ϕ cannot be contracted to a point in L^+_q. Therefore, if $\phi \subset L^+_q$ contains y in its interior, then it represents a non-trivial class of the homology group of the space L^+_q. Moreover, if some other closed curve $\phi' \subset L^+_q$ is homotopy equivalent to ϕ (and therefore, corresponds to the same element in the first homology group), then ϕ' is also a high potential fence caging the sheep at the point y. Thus, it is enough to consider one closed curve per first homology class in L^+_q to check if L^+_q has a closed curve whose interior contains y.

Let us now interpret the shape of the set L^+_q geometrically. Since by definition any potential function strictly monotonically decreases with the distance to the set of robots, for any $q > 0$ the set L^+_q is in fact a union of closed balls of radius R centred at points $\{x_1, x_2, \ldots, x_n\}$, where $p_d(R) = q$: $L^+_q = \bigcup_{i=1}^n B_R(x_i)$.

This observation is crucial, as it enables us to consider discrete approximations of L^+_q without losing any important information about the topological properties of the latter. Namely, consider an alpha complex $A(R)$ (see Sect. 4 for the introduction to alpha complexes). It is well known (Edelsbrunner and Harer 2010) that $A(R)$ is homotopy equivalent to and lies strictly inside of $\bigcup_{i=1}^n B_R(x_i)$, and therefore preserves its topological properties of interest. Moreover, each first homology class representative of $A(R)$ whose interior contains y is a caging chain, as it consists of the vertices corresponding to the positions of robots, and links between them. Thus, the first homology group of $\bigcup_{i=1}^n B_R(x_i)$, can be extracted directly from $A(R)$.

For each first homology class representative[4] g of $A(R)$ we check if y lies inside its interior. If this is the case, then the sheep is caged by means of the caging chain g.

Recall from Sect. 5 that we define the potential of the fence as the potential of its weakest point, $p_c(\phi) = min_{x\in\phi}(p_c(x))$. Let us now define *an optimal* high potential fence:

Definition 6 Given a configuration c of robots, an *optimal potential fence* is a fence of the highest possible potential.

To find an optimal potential fence, we need to find the *maximum* value $q_{max} > p_c(y)$ for which there exists $\phi \subset L^+_{q_{max}}$ containing y in its interior. For that, we consider a family of sets $\mathcal{F} = \{L^+_{q_0}, L^+_{q_1}, \ldots, L^+_{q_k}\}$, where $\infty = q_0 > q_1 > \cdots > q_k$ and hence $L^+_{q_0} \subset L^+_{q_1} \subset \cdots \subset L^+_{q_k}$. Here the family $\mathcal{F}$ is finite, as explained in Sect. 4.

Then q_{max} is the greatest value among $q_0 > q_1 > \cdots > q_k$ such that $L^+_{q_{max}}$ contains a closed curve whose interior contains y.

Let us now make an important observation:

Proposition 1 *The caging chains computed as first homology class representatives of the corresponding alpha complex are optimal.*

Proof Let the sheep be located at point y, and the robots be at a configuration c. Let the sheep be caged, and let q_{max} be the potential value of the optimal high potential fence caging the sheep. Then, there is a closed curve $\phi \subset L^+_{q_{max}}$, such that $y \in \text{int}(\phi)$. Recall that $L^+_{q_{max}} = \bigcup_{i=1}^n B_R(x_i)$, where $p_d(R) = q_{max}$.

Consider an alpha complex $A(R)$. Since $A(R)$ is homotopy equivalent to $\bigcup_{i=1}^n B_R(x_i)$, there exists a closed curve $\phi' \simeq \phi, \phi' \subset A(R) \subset \bigcup_{i=1}^n B_R(x_i)$. Therefore, ϕ', and any other closed curve from $A(R)$, homotopy equivalent to it, is an optimal high potential fence caging the sheep. □

[4] We compute homology with coefficients in $\mathbb{Z}_2$.

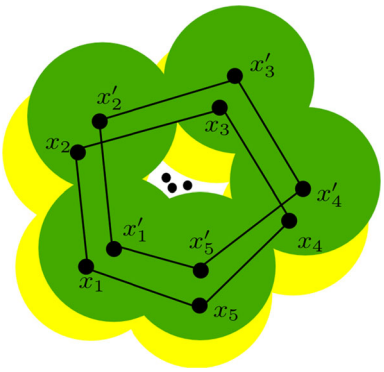


Fig. 5 This figure illustrates a typical motion of the robots during herding phase. The robots moves from $(x_1, x_2, x_3, x_4, x_5)$ to $(x'_1, x'_2, x'_3, x'_4, x'_5)$. Yellow circles depict the initial superlevel set, while the final one is depicted in green (Color figure online)

Algorithm 3: Cage verification

input : robots coordinates $(x_1, \ldots, x_n)$, sheep coordinate y
output: $True$ (if robots form a cage with sheep inside) or $False$

$D \leftarrow$ Delaunay$((x_1, \ldots, x_n))$
foreach $L_q^+ \subset D$ *such that* $q > p_c(y)$ **do**
  $L_q^+ \leftarrow$ AlphaComplex$((x_1, \ldots, x_n))$
  $\mathcal{G}_1 \leftarrow$ FirstHomologyRepresentatives(L_q^+)
  foreach $g \in \mathcal{G}_1$ **do**
    if *IsInsidePolygon(y, g)* **then**
      return $True$
    end
  end
end
return $False$

So, to check if the sheep with coordinates y is caged, we perform the following procedure, see Algorithm 3. We construct a Delaunay triangulation $D(\{x_1, \ldots, x_n\})$ based on the coordinates of the robots. Then, we build a sequence of superlevel sets (represented as alpha complexes) $\mathcal{F} = \{L_{q_0}^+, L_{q_1}^+, \ldots, L_{q_k}^+\}$, where $q_k > p_c(y)$. Note that the number of superlevel sets is finite, as only the values q_i that change the number of simplices in the respective superlevel sets are considered. If there is such a superlevel set $L_{q_{cage}}^+$, in which one of the first homology class representatives contains y in its interior, then the sheep is caged. If there are several such superlevel sets, we select the one with the largest potential value. In practise, this is the same as to construct a nested sequence of unions of closed balls with growing radii, starting with $R = 0$ and finishing once $R \geq \mathrm{d}(y)$. In its turn, the union of balls can be replaced with the corresponding alpha complex.

8 Safely connected cages

Recall that the sheep can move in arbitrary directions with a constant speed during the herding phase. Therefore, we need to derive some sufficient conditions to guarantee that the sheep will not escape from the robots as they move between two caging configurations c_1 and c_2. If the robots can move between two configurations in this way, we call c_1 and c_2 *safely connected*. The following proposition tells us how the robots can move during the herding phase without letting the sheep escape far from them, see Fig. 5.

Proposition 2 *Consider a potential function $p_c : \mathcal{W} \to \mathbb{R}$. Assume that the robots move from a caging configuration c_0 to another configuration c_T, and let $c(t)$ denote their trajectory in $\mathcal{C}_{free}$, $t \in [0, T]$. Let s_t denote the trajectory of a sheep in $\mathcal{W}$, and assume that when $t = 0$ the sheep is caged by robots $\{r_1, \ldots, r_k\} \subseteq \mathcal{R}$. Finally, let g_0 be a caging chain formed by $\{r_1, \ldots, r_k\}$, $s_0 \in \operatorname{int} g_0$, and let g_t denote its deformation at time t, induced by the corresponding movement of the robots.*

Then the sheep never escapes the cage for any $t \in [0, T]$, provided that

1. *both the sheep and the robot team move with constant non-zero speeds, $\mathrm{v_s}$ and $\mathrm{v_r}$;*
2. $\mathrm{d}_{c(0)}(s_0) - \omega(g_0) > (\mathrm{v_s} + 2 \cdot \mathrm{v_r}) \cdot T$

Proof We need to demonstrate that for any moment of time $t \in [0, T]$

- the potential of the sheep is lower than the potential of the caging chain, $p_{c(t)}(s_t) < p_{c(t)}(g_t)$, and
- the sheep is inside the caging chain, $s_t \in \operatorname{int} g_t$.

Note that the latter statement follows from the former, provided $s_0 \in \operatorname{int} g_0$. Therefore, it is enough to show that for any $t \in [0, T]$

$$p_{c(t)}(s_t) < p_{c(t)}(g_t). \tag{2}$$

From the first condition of the Proposition, for any $t \in [0, T]$ we have

$$\mathrm{d}_{c(t)}(s_t) \geq \mathrm{d}_{c(0)}(s_0) - t \cdot \mathrm{v_s} - t \cdot \mathrm{v_r}.$$

Therefore, since p strictly monotonically decreases with the distance to the robots, we have

$$p_{c(t)}(s_t) \leq p_d(\mathrm{d}_{c(0)}(s_0) - t \cdot \mathrm{v_s} - t \cdot \mathrm{v_r}).$$

Similarly, for $t \in [0, T]$ we have

$$p_{c(t)}(g_t) \geq p_d(\omega(g_0) + t \cdot \mathrm{v_r})$$

To prove Eq. (2), it is enough to show that

$$p_d(\omega(g_0) + t \cdot \mathrm{v_r}) > p_d(\mathrm{d}_{c(0)}(s_0) - t \cdot \mathrm{v_s} - t \cdot \mathrm{v_r}), \quad (3)$$

which by strict monotonicity of $p()$ follows from

$$\mathrm{d}_{c(0)}(s_0) - t \cdot \mathrm{v_s} - t \cdot \mathrm{v_r} > \omega(g_0) + t \cdot \mathrm{v_r}, \quad (4)$$

which can be written as

$$\mathrm{d}_{c(0)}(s_0) - \omega(g_0) > t \cdot \mathrm{v_s} + 2 \cdot t \cdot \mathrm{v_r}.$$

The latter is true for any $t < T$, since

$$\mathrm{d}_{c(0)}(s_0) - \omega(g_0) > (\mathrm{v_s} + 2 \cdot \mathrm{v_r}) \cdot T$$

□

Remark 1 Instead of expressing these conditions in terms of time and speed, we can as well assume that the time T is fixed. This gives us a condition on the acceptable displacement of the robots, and we then can adjust the speed $\mathrm{v_r}$ to satisfy the necessary conditions from the above proposition. Let $\varepsilon_r = \mathrm{v_r} \cdot T$ and $\varepsilon_s = \mathrm{v_s} \cdot T$. In our implementation, we make sure that $\mathrm{d}_{c(0)}(s_0) - \omega(g_0) > \varepsilon_s + 2 \cdot \varepsilon_r$. Assuming that the speed of robots is higher than the speed of sheep, it is enough to check that $\mathrm{d}_{c(0)}(s_0) - \omega(g_0) > 3 \cdot \varepsilon_r$.

9 Motion planning

Consider now the steering problem. Assume that we have an initial caging configuration c_{init}. We need to move the flock to the specified goal region without breaking the cage. This means that each configuration in the path must form a cage, and all the subsequent pairs of cages must be safely connected. Let U be a set of all possible actions which we can apply to a given configuration in order to move to the next one. Namely, each $u \in U$ specifies a collection $\{\mathbf{v_1}, \ldots, \mathbf{v_n}\}$ of vectors defining a movement of the system. When u is applied, the ith robot moves from x_i to $x_i + \mathbf{v_i}$. For simplicity, we discretize U by assuming that each robot can be moved along some vector $\mathbf{e_{k\Delta\phi}}$ at a distance $||m\Delta d||$, where $\Delta\phi$ and Δd are the discretization steps of vector orientation and translation, respectively. By $\mathcal{C}_{cage} \subset \mathcal{C}_{free}$ we denote the subspace containing all caging configurations.

Our goal is to demonstrate that herding by caging can be integrated with existing sampling-based motion planning algorithms. For this purpose, we consider standard Rapidly-exploring random tree (RRT) algorithm (LaValle and Kuffner 2001). The herding by caging algorithm does not need to specifically rely on RRT and can be combined with other sampling-based motion planners: our contribution lies in proposing the procedure for sampling novel configurations. Since most sampling-based path planners have the same high level structure, we used basic RRT to illustrate it. The main goal of the experiments was therefore not to test a particular sampling-based path planning algorithm, but rather to illustrate that it is possible to sample caging configurations of different shapes in order to avoid obstacles and pass through narrow passages.

9.1 Standard RRT: a naive approach

We start with a simple approach to motion planning—standard RRT with rejection sampling. We sample random configurations from $\mathcal{C}$, and check whether they are (i) collision-free, and (ii) form valid cages of the required size. If both conditions hold, we then would search for the nearest vertex in the RRT, and make a motion from it towards the randomly selected configuration, so that the nearest and the new vertices would be (i) reachable from each other by a straight collision-free motion, and (ii) safely connected. However, this approach does not perform well in practice, as the probability of randomly sampling a caging configuration in $\mathbb{R}^{2n}$ is small. Therefore, instead of sampling random configurations from $\mathcal{C}$, we aim to sample new configurations biased towards the caging subset of the configuration space, $\mathcal{C}_{cage}$, see Algorithm 4.

Algorithm 4: RRT-based caging planner

```
input  : configuration space C, robots initial configuration c_init,
         sheep coordinates f, workspace W
output: RRT T
T ← ∅
if CageVerification (c_init, f) then
    T ← {c_init, f}
    t_start ← CurrentTime()
    while time < MaxTime do
        c, f ← RandomFromTree(T)
        δ ← Rand(0, 1)
        if δ < 0.5 then
            c_rand ← RandomTranslation(c)
        else
            c_rand ← RandomPerturbation(c)
        end
        if Extend(T, c_rand, W) = Reached then
            break
        end
        time ← CurrentTime() - t_start
    end
end
return T
```

9.2 RRT-based caging planner

Namely, we generate samples based on those caging configurations which we already have. We randomly choose an

existing configuration c from the RRT, and with probability 0.5 we apply one of the following operations to generate new configuration c_{rand}: *Random translation:* in this case, c_{rand} has the same shape as c (i.e., relative positions of the robots are preserved), but is moved from c in a random direction to random distance. *Random perturbation:* in this case, we randomly choose a direction $\phi_i \in (0, 2\pi)$ for each robot, $i \in \{1, \ldots, n\}$, and move the ith robot in the chosen direction at a distance $\varepsilon(c)$, where $\varepsilon(c)$ depends on the location of the closest sheep and the width of the caging chain, and is defined in such a way that c_{rand} is also a caging configuration.

Once we have a new random configuration c_{rand}, we are trying to extend the RRT, see Algorithm 5. For that, we find the nearest existing configuration c_{near} in the tree, and then with probability $1 - \rho$ we apply the action from U that leads us as close as possible to c_{rand}. Alternatively, with probability $\rho > 0$ we apply to c_{near} a random action instead of the optimal one. This is done for the sake of preserving probabilistic completeness in our modification of RRT. However, in practice our algorithm is more efficient when ρ is close to 0.

Algorithm 5: Extend

input : RRT $\mathcal{T}$, configuration c_{rand}, workspace $\mathcal{W}$
output: *Reached* or *Extended* or *Failed*

$c_{near}, f \leftarrow$ NearestNeighbor($\mathcal{T}, c_{rand}$)
$\delta \leftarrow$ Rand(0, 1)
if $\delta < \rho$ **then**
  $c_{new}, f' \leftarrow$ ApplyRandomAction($\mathcal{T}, c_{near}, f, c_{rand}$)
else
  $c_{new}, f' \leftarrow$ ApplyTheBestAction($\mathcal{T}, c_{near}, f, c_{rand}$)
end
if *AddVertex*($\mathcal{T}, c_{new}, f', c_{near}, f, \mathcal{W}$) **then**
  if $c_{new} \in \mathcal{C}_{goal}$ **then**
    return *Reached*
  else
    return *Extended*
  end
else
  return *Failed*
end

Algorithm 6: AddVertex

input : RRT $\mathcal{T}$, configuration to be added c_{new}, sheep coordinates corresponding to the new configuration f', nearest configuration in the tree c_{near}, sheep coordinates within nearest configuration f, workspace $\mathcal{W}$
output: *True* or *False*

$\mathcal{M} \leftarrow$ GetLinearMotion($c_{near}, f, c_{new}, f', \mathcal{W}$)
$c_{new}, f' \leftarrow$ LastBeforeCollision($c_{near}, f, \mathcal{M}, \mathcal{W}$)
return *CageVerification*(c_{new}, f') **and** (*Distance*(c_{near}, c_{new}) $> \varepsilon$) **and** *AreSafelyConnected*(c_{near}, c_{new}, f)

Algorithm 7: AreSafelyConnected

input : Two configurations c_0 and c_1, sheep coordinates s_0 corresponding to the first configuration
output: *True* or *False*

$\Delta \leftarrow$ GetMaxRobotMovement(c_0, c_1)
return *DistanceToConf*(s_0, c_0)$-$*ConfChainWidth*(c_0) $> 3\Delta$

9.2.1 Motion verification during the herding phase

As a result of applying an action to c_{near}, we get a new configuration c_{new} which we would like to add to the tree. At this point we need to make sure that c_{near} and c_{new} are safely connected, and there is a straight-line collision-free path between them, see Algorithm 6. If there is no straight-line collision-free path, we compute the closest reachable configuration on the way to c_{new} instead. If it is a caging configuration and lies at a sufficient distance from c_{near}, and c_{near} and c_{new} are safely connected (by Proposition 2), we add it to the tree.

The correctness of our approach is enforced by two verification procedures: *CageVerification*(c_{new}, f') and *AreSafely Connected*(c_{near}, c_{new}, f). The former corresponds to the caging verification during the repulsion phase and is described in details in Algorithm 3. The latter checks the safety during the herding phase, see Algorithm 7. Here, the function *GetMaxRobotMovement*(c_0, c_1) computes ε_r from Remark 1—the length of the paths between the initial and the resulting positions of the robots. *DistanceToConf*(s_0, c_0) returns $d_{c_0}(s_0)$, and *ConfChainWidth*(c_0) returns $\omega(g_0)$.

9.2.2 Probabilistic completeness

Let us now discuss probabilistic completeness of the proposed algorithm. Assume that there is a sequence of configurations $c_1, \ldots, c_q$, such that $c_1 = c_{init}$, and $c_q \in \mathcal{C}_{goal}$. Assume also that there exists a sequence of actions $u_1, \ldots, u_{q-1}$ that when applied to c_1 yields the sequence $c_2, \ldots, c_q$. Here all of the configurations are in the same connected component of $\mathcal{C}_{cage}$, and each pair of subsequent configurations are safely connected.

Then we can formulate the following proposition (the proof is almost identical to the proof of Theorem 3 from the work by LaValle and Kuffner (2001), and we briefly recall it here).

Proposition 3 *The probability that the above proposed algorithm initialized with c_{init} will contain a configuration from $\mathcal{C}_{goal}$ tends to one as the number of iterations goes to infinity.*

Proof Assume the RRT contains c_i as a vertex after some finite number of iterations. Consider the Voronoi diagram associated with the RRT vertices. Since in our implementation no two RRT vertices lie within a specified $\varepsilon > 0$ of each other, the measure of the Voronoi cell $\mu(\text{Vor}(c_i)) > 0$. At

Fig. 6 In our experiments, we use 8 mobile robots to act as herders (black) and 3 mobile robots to act as sheep (yellow). All the robots have the same radius of 0.12 m (Color figure online)

each iteration of the algorithm, there is a non-zero probability p_1 that c_i will be selected as the nearest vertex. Since U is finite, and we select a random action from U with positive probability ρ, there is a non-zero probability p_2 that the right action u_i leading us to the new configuration c_{i+1} will eventually be selected. We apply this argument iteratively from c_1 to c_{q-1}. □

10 Experiments with mobile robots

In this work, we first evaluate the proposed herding system using 11 low-cost mobile robots shown in Fig. 6. Each robot is equipped with a three-wheeled omnidirectional mobile platform, a Raspberry Pi 3B single-board computer, as well as a LED light to show its real-time motion phase. All the robots are localized using the VICON motion capture system[5] During operation, the sheep robots can acquire the locations of all herder robots, so as to react to the potential fields generated by the herders during the repulsion periods.

We set a constant speed for herders as $v_r = 0.2$ m/s. The sheep robots can operate in either the *wander* mode or the *escape* mode. During the herding phase, the sheep move with a constant speed of $v_s^H = 0.08$ m/s randomly when operating in the wander mode, while they always move in the direction which is opposite to the target area when operating in the escape mode. During the repulsion phase, all the sheep move in the opposite direction of their respective closest herder and the speed of each sheep is determined by the potential:

$$p_i = 0.1^{d_i}, \quad 1 \leq i \leq m \tag{5}$$

where $p_i \in \mathbb{R}^+$ is the potential of the i-th sheep and $d_i \in \mathbb{R}^+$ is its distance to the closest herder. Denoting the maximum speed of the sheep $v_s^R = 0.2$ m/s, the speed of the i-th sheep under repulsion is then calculated by the function:

$$v_s^r = \frac{v_s^R}{1 + e^{\frac{1}{-d_i}}} \tag{6}$$

Unless otherwise stated, the aforementioned speeds will be used in all experiments.

The experiment workspace has the size of 8 m × 5 m. As will be seen shortly, we designed 6 different maps for the experiments considering the feasibility for the scales of herding teams. The experiments conducted in this work focus on evaluating: (a) the initial cage acquisition algorithm given different number of sheep with different velocities; (b) the path planning algorithm's performance in terms of the numbers of herders and sheep; (c) the effect of team deformation when the herding team passes through narrow passages; and (d) the effect of team reformation in case any herder robot malfunctions during the herding process.

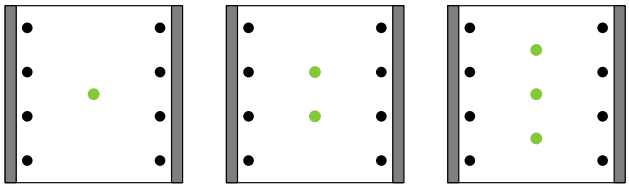


Fig. 7 The experiment is conducted in an empty space. The initial positions of all standby herders (black) and sheep (green) are shown (Color figure online)

Table 1 Minimal number of herders to form cage

#Herder Ratio	# Sheep		
	1	2	3
0.10	3	4	4
0.25	4	4	6
0.40	4	6	8

10.1 Initial cage acquisition

The initial cage acquisition is essential to enable the herding process. In this work, given different number of sheep with different wandering velocities, we evaluate the minimal number of needed herders to form an initial cage, as well as how the herders should move to achieve the initial cage. In this experiment, we keep $v_r = 0.2$ m/s and vary the ratio of v_s^H / v_r.

As shown in Fig. 7, we assume that the standby herders and the sheep to be caged are located in an empty space. By setting different speed ratios, we aim at evaluating the minimal number of needed herders to form initial cages. The result of the proposed initial cage acquisition algorithm is reported in Table 1. We can see that only 3 out of 8 herders were needed

[5] https://www.vicon.com/.

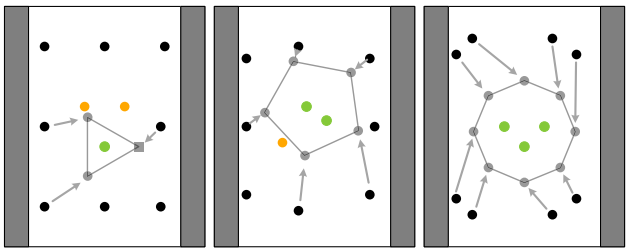


Fig. 8 Initial cage acquisition path planning. Green dots mark the target sheep to be caged. Black dots are the initial positions of all the 8 standby herders, and gray dots are the positions of the selected herders when an initial cage is achieved (Color figure online)

when the ratio was 0.1 and there was only 1 sheep. When the ratio increased or the number of sheep became larger, more herders were needed to form the initial cages. This is due to when the sheep can move faster or if there are more sheep, the potential locations of the sheep will be distributed in a larger area, for which we need more herders to safely form the initial cages.

Some additional evaluations are shown in Fig. 8. In this experiment, the initial positions of the sheep are randomized within a small range. We target on 1, 2 or 3 sheep with respective speed ratio of 0.1, 0.25 or 0.4 for acquiring initial cages. In the 3 shown scenarios, 3, 5 and 8 herders were needed and we can see that the selected herders were relatively close to their assigned caging positions. This allows the herders to form the caging configurations as quickly as possible in order to avoid the sheep to further expand their potential distribution, so as to minimize the number of needed herders.

10.2 Sheep movement modes and repulsion time

In wander or escape modes, sheep behave differently in terms of their intention of escaping from the cage during the herding phase, in which their movement do not follow the potential field. When operating in the wander mode, sheep move randomly and their potential will accordingly be randomly increasing or decreasing. However, since in escape mode sheep will always move towards the closest edge of the cage, their potential will be monotonically increasing during the herding phase, and therefore increase their probability of escaping from the cage.

Additionally, denoted by t_H the time for each herding phase and t_R the time for each repulsion phase. While switching between herding and repulsion phases during the process, setting different ratios of t_R/t_H will result in different movement constraints for sheep. Intuitively, a larger ratio will give the herders more time to constrain the movement of sheep by exerting the potential fields. In contrast, a smaller ratio will allow the sheep more time to move either randomly or move in the escaping direction, which will increase the potential during the whole process.

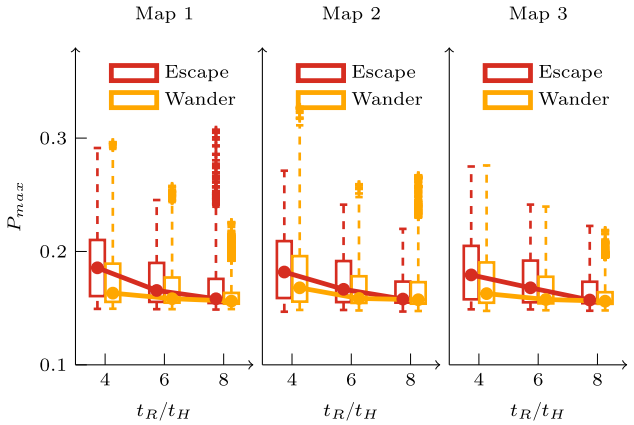


Fig. 9 Statistics of the maximum potential of sheep in terms of different movement modes and ratios of t_R/t_H

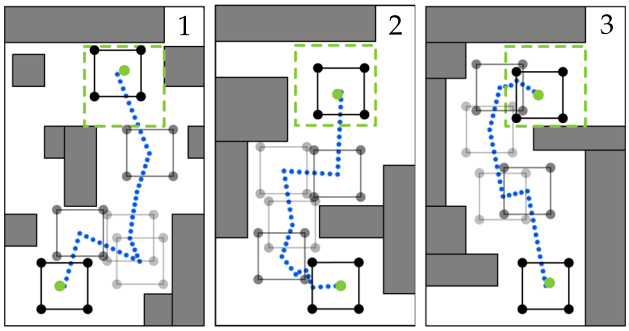


Fig. 10 Herding experiment in Map 1, 2 and 3. The black and gray dots denote the sampled locations of herders during the process, the green dots depict the locations of the sheep, and the green dash lines indicate the target sheep . The path of the geometric center of the herding team is shown in blue. The maximum potential of the sheep has been sampled against all herders and were recorded during the whole process (Color figure online)

In order to understand the effects given by the sheep movement modes and the ratio of t_R/t_H, we ran experiments by setting different values for them and report the resulted statistics in Fig. 9. In this experiment, we assume the initial caging configurations are already achieved and we use 4 herders to herd 1 sheep and repeat the same path in each of the 3 maps depicted in Fig. 10. As seen in Fig. 9, with different ratios of t_R/t_H, the mean of P_{max} values of the escape mode were always higher than the wander mode as expected. Furthermore, escape mode presented the variation ranges of P_{max} always larger than the wander mode. This is due to the sheep operating in the wander mode tend to move around a small area. In the escape mode, however, the sheep would always move towards different closest caging edges, resulted in larger movement regions relative to the caging team, and hence generated larger variations in P_{max}.

Observe in Fig. 9 that, when operating in the wander mode, the boxplots show more outliers than that of operating in the escape mode. This is because the P_{max} values of the wander

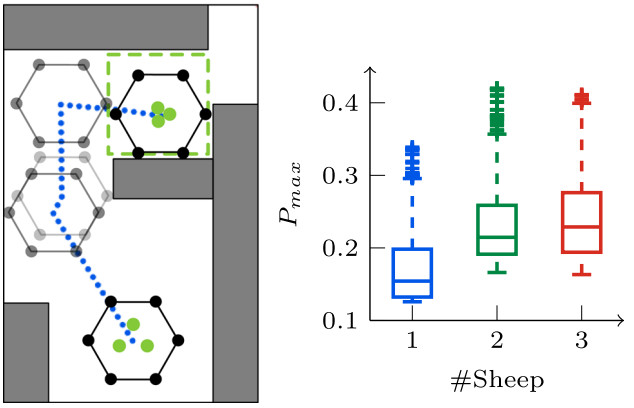


Fig. 11 Herding experiments with different number of sheep. Left: The herding path generated by the algorithm for 6 herders. Right: The maximum potential of the sheep has been sampled against all herders and were recorded during the whole process

mode were more concentrated in a small range, and therefore more data points which are not close to the mean were considered outliers. This phenomenon can also be noticed when comparing the outliers across different ratios of t_R/t_H, for which a larger value resulted in more concentrated data distribution around the mean, and hence more data points were considered as outliers.

10.3 Number of herders and sheep

In this experiment, we evaluate the herding procedure in terms of the numbers of herders and sheep. To keep the evaluation consistent, we set the ratio of $t_R/t_H = 4$ and set the sheep to operate always in the wander mode. Also, we assume the initial cages are already acquired using the proposed algorithm.

As seen in Fig. 11, in order to compare the effect given by different number of sheep, we did not minimize the number of herders in this test, and instead always use 6 herders to herd 1, 2 and 3 sheep in map 4. For this, we initially ensured that all the sheep will be caged using this configuration, and repeat the same herding path for different tests. We can see from the statistics that the P_{max} values were increasing when more sheep were herded. This is because, for the same herding path, more sheep within the cage constrained the movement ranges of each other, and the central area in the cage is not available for any of the sheep. Therefore, the sheep tend to move closer to the herders resulted in higher potential.

For evaluating the impact of the number of herders, we evaluated the herding procedure using 4–8 herders with 1 sheep in map 5. As depicted in Fig. 12, due to the shape of the herding teams were different for different number of herders, the herding paths were accordingly different to ensure shape specific collision-free paths. The statistics reported in Fig. 12 show that the P_{max} values were increasing while more herders were involved in the task. This does not mean that more herders would make the herding procedure more risky. In contrast, more herders would increase the potential fields surrounding the sheep to given more constraints to the sheep's movements. Therefore, although the P_{max} were higher when using more herders, it is in fact more robust for herding.

10.4 Team deformation

After an initial caging configuration is acquired, the herding team has to pass through different structures in the map to reach the destination. During this procedure, there can be narrow passages which the herding team cannot pass without collision if it does not deform to adapt to the shape of the narrow passage. In order to achieve this, the proposed planner (Algorithm 4) introduced a random perturbation term for adjusting the shape of the herding team during path planning. As such, the planning algorithm is able to deform the herding team while still ensuring a caging configuration.

In this experiment, we use 4 herders to herd 1 sheep in map 6, in which we designed a narrow passage to force team deformation. As seen in Fig. 13, the herding team was not able to directly pass through the narrow passage. As such, it was gradually deformed into narrower shapes to fit through, and then resumed to a normal shape to finally reach the destination.

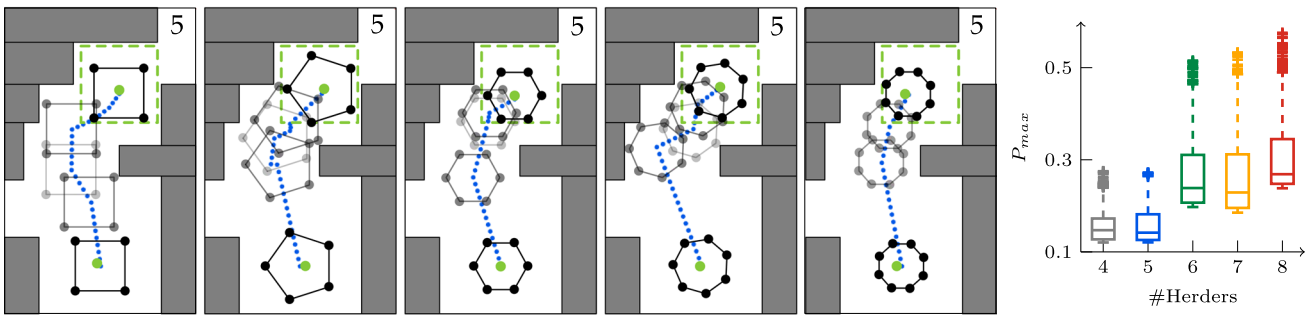


Fig. 12 Herding experiments with different number of herders. Left: Herding path for 4–8 herders. Right: The maximum potential of the sheep has been sampled against all herders and were recorded during the whole process

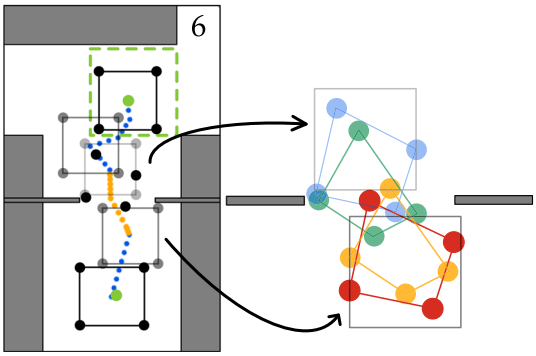


Fig. 13 Herding experiment with team deformation. When passing through the narrow passage, the team deformation process is depicted by a close-up figure on the right. In the deformation procedure, the shape of the herding team was changing in the order of red, yellow, green and blue, and then resumed to the normal shape (Color figure online)

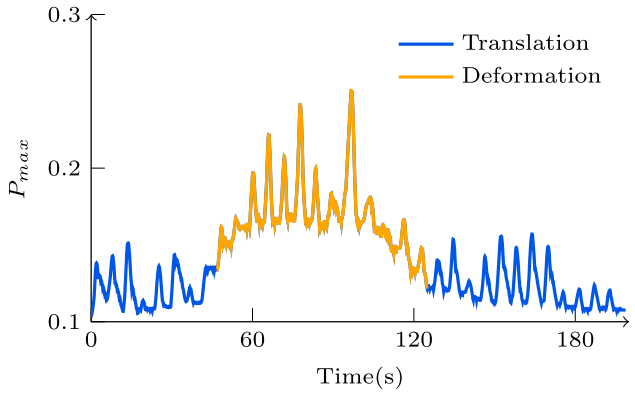


Fig. 14 The maximum potential of the sheep has been sampled against all herders and were recorded during the whole process

The P_{max} values during this herding procedure is reported in Fig. 14. We can observe that the P_{max} values were relatively small and varied in a small range. Once the deformation started, the P_{max} values significantly increased and started oscillating largely. Thereafter, the P_{max} returned to a small range after the team shape resumed to the normal situation. This result shows that the proposed path planning algorithm is able to handle the cases where team deformation is required, while ensuring that the caging configurations will be kept to finally achieve the task.

10.5 Team reformation

During a herding task, it is possible that one or more of the herders will be broken or malfunctioning in the middle of the procedure. In those cases, instead of using the minimum number of herders, if we in the beginning assigned redundant herders in the team, we can still complete the task by reforming the team at the time of failures. In this experiment, we set the sheep to operate in the wander mode, and conducted 2 tests with redundant herders: (1) 7 herders to herd 2 sheep and (2) 5 herders to herd 1 sheep. During the herding process, we artificially designate one herder malfunction forcing it to leave the team, and then we trigger the initial cage acquisition algorithm to reform the team shape in order to continue the task.

The experiment results are reported in Fig. 15. We can observe that the P_{max} values before and after the reformations were similar, which demonstrated a similar effect as in the previous experiments in Fig. 12. Note that in both tests, there were a sudden increase of the P_{max} values, which was caused by the malfunctioning of one of the herders which introduced a less constrained system during that short period.

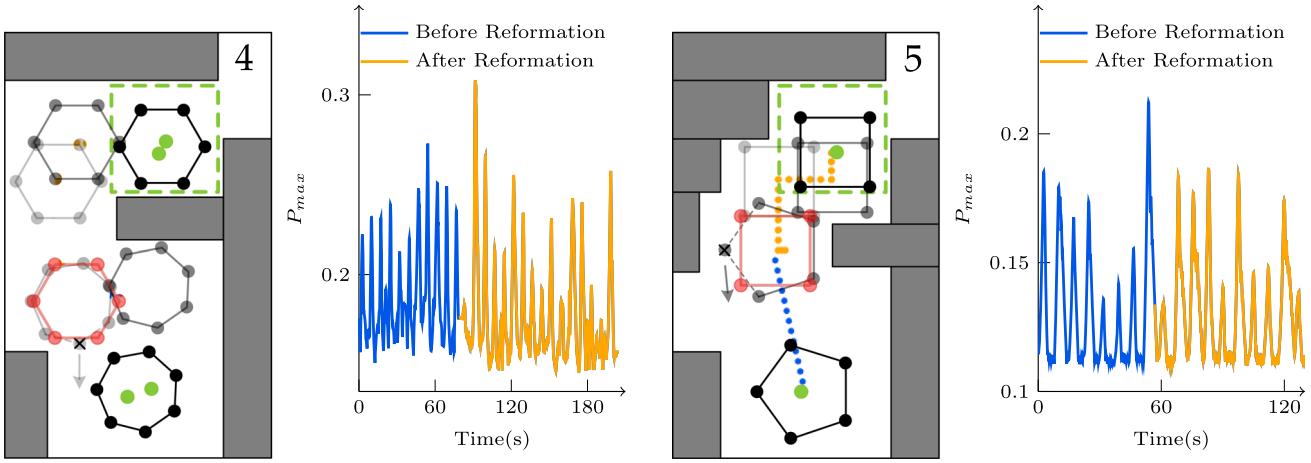


Fig. 15 Herding experiments with team reformation caused by the failure of 1 herder during the process. The shape reformation processes are depicted in red. Left: 7 herders with 2 sheep. Right: 5 herders with 1 sheep (Color figure online)

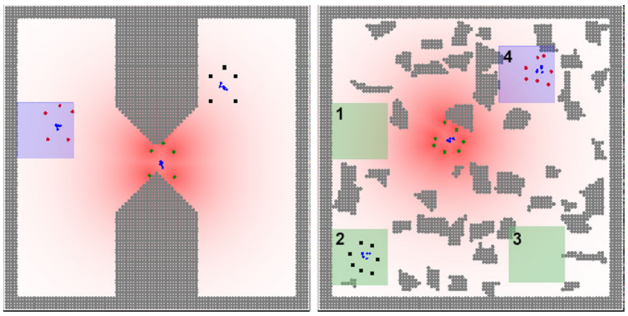


Fig. 16 The obstacles are depicted in grey; the goal region is the blue square; the black dots correspond to the initial configuration of the robots, the green dots depict the current configuration, and the red dots are the robots in the final configuration. The potential field induced by the current configuration is depicted in red. In the second environment, the 3 remaining goals are depicted in green (Color figure online)

11 Experiments in simulated environments

In this section, we present our initial experimental results obtained in simulations. Here, we investigate how our herding algorithm performs in cluttered environments and in the presence of narrow passages.

Currently, no finite time convergence guarantees for sampling-based motion planners, such as RRT, exist in literature (Dantam et al. 2018). This means that it is generally impossible to exactly determine whether there is no path between the start and the goal, or if the planner needs more iterations to find it. Thus, in practice, it is common to rely on heuristic stopping criteria, such as a time out (30 s in our experiments) or the maximum number of samples.

In our experiments, we want to investigate whether our algorithm is able to compute paths in the presence of narrow passages and randomized cluttered environments. For this, we construct workspaces with narrow passages of different width, and experimentally evaluate the success rate of the planner. We use the simulated environment to quantitatively evaluate the computation time and the success rate of our algorithm, as such complex scenes would be difficult to construct in our indoor environment.

We consider two types of workspaces and run our algorithm with the different number of robots. We run our experiments on an Intel i7 CPU laptop with 12 GB of RAM.

11.1 Narrow passages

In our first experiment, we consider a family of simple rectangular workspaces containing two polygonal obstacles and a narrow passage between them (Fig. 16, left side). Our goal is to see how the efficiency of our algorithm depends on $d(\mathcal{R})/w(\mathcal{C})$, where $w(\mathcal{C})$ denotes the width of the narrow passage, and $d(\mathcal{R})$ is the maximal distance between any two robots in the initial configuration. In other words, we would like to see how well the robots can change the shape of their formation without breaking the cage, when they have to go through narrow passages. We perform experiments with 4, 5, 6, and 7 robots respectively. We consider 20 random initial configurations and run the algorithm 50 times for each of them. We interrupt it after 30 s in case the solution has not been found. We compute the average execution time in seconds, as well as the success rate (the percentage of successful runs). Since this workspace is very simple, and the main difficulty is to find the path through the narrow passage, we consider only one goal region, and all the initial configurations are located on the opposite side of the workspace.

Table 2 The average computation time of the successful runs and the success rate in the presence of narrow passages

$\frac{d(\mathcal{R})}{w(\mathcal{C})}$	4 robots	5 robots	6 robots	7 robots
1.0	1.7 s, 96%	2.0 s, 91%	3.1 s, 89%	4.3 s, 79%
0.9	2.8 s, 89%	2.9 s, 87%	1.5 s, 77%	4.3 s, 76%
0.8	2.9 s, 79%	3.3 s, 86%	1.8 s, 57%	2.7 s, 71%
0.7	3.7 s, 73%	3.5 s, 62%	3.3 s, 44%	9.9 s, 44%
0.6	7.1 s, 45%	2.9 s, 26%	3.5 s, 20%	8.6 s, 26%

Table 3 The average computation time of the successful runs and the success rate in randomized cluttered environments

Goal	4 robots	5 robots	6 robots	7 robots
1	6.9 s, 87%	7.5 s, 91%	7.3 s, 80%	6.5 s, 86%
2	7.0 s, 87%	5.2 s, 90%	10.5 s, 69%	9.9 s, 96%
3	17.1 s, 16%	5.2 s, 1%	6.3 s, 2%	5.9 s, 2%
4	2.9 s, 96%	5.3 s, 90%	7.4 s, 89%	4.7 s, 91%

The Table 2 presents the result of the experiment. We see that for $d(\mathcal{R})/w(\mathcal{C})$ between 0.8 and 1.0 our algorithm performs quite well. However, we can observe that as we decrease the width of the passage, the success rate significantly decreases. This likely is the case because our algorithm cannot explore the caging space well enough within 30 seconds, as the shapes of the new configurations are biased towards those which are already in the tree.

11.2 Random polygons

In the second experiment, we consider a more complex environment (Fig. 16, right side). As before, we analyze how well our algorithms perform depending on the number of robots: 4, 5, 6, and 7. We consider 4 different goal regions and 20 random initial configurations. We run the algorithm 50 times for each combination of the start and the goal. We interrupt it after 30 s in case the solution has not been found.

The Table 3 presents the result of the experiment. This experiments shows the performance of our algorithm in a

cluttered environment. Here, we can see that the Goal 3 is practically unreachable within 30 s. This happens because it is densely surrounded with obstacles, and therefore the shape of the configurations in the path should be significantly different from the shape of the initial one.

12 Conclusion and future work

In this paper, we extend our previous work and present an approach towards *herding by caging*: given a set of mobile robots in the two-dimensional workspace, we partially control the set of moving objects (called sheep) by means of the potential field induced by repulsive forces exerted by the robots. To guarantee that the sheep cannot escape from the herders, we formalize the problem using the concept of *potential-based caging*.

We present cage acquisition and verification algorithms. The caging verification problem is addressed using a classical tool from algebraic topology—homology groups. Furthermore, we propose an RRT-based algorithm for path planning and show that it preserves an important advantage of RRT—probabilistic completeness. We implement our algorithms and analyze their performance in simulated and real-world experiments.

Currently, we assume that the robots are controlled in a centralized way: the positions of all robots are known to the central planning algorithm. In the future, it would be interesting to relax this assumption and extend the method to the case where the robots need to make decisions in a de-centralized way and address the challenges arising from limited communication between them.

In the future, we plan to extend the work to the case of 3-dimensional workspaces. Our theoretical framework can be generalized to this case, as instead of 1-homology generators capturing holes we can consider 2-dimensional generators, capturing voids. As before, a potential function $p : \mathcal{W} \to \mathbb{R}$ has to be continuous and strictly monotonically decreasing as the distance from the point to the set of robots increases, and the same assumption regarding the motion of the flock is made. A high potential fence can be considered as a high potential subset HPS of $\mathcal{W}$, such that at least one connected component of $\mathcal{W} - HPS$ is bounded. Proposition 1 could then be generalized, as the dimensionality of the space does not play a crucial role in the proof. The implementation and experiments, however, will require some additional work and technical considerations.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s10514-021-09975-8.

Acknowledgements This research work was supported by the Innovation and Technology Fund of the Government of the Hong Kong Special Administrative Region (Project No. ITS/104/19FP), Knut and Alice Wallenberg Foundation, Swedish Research Council, and European Research Council (Project 884807 BIRD).

References

Bacon, M., & Olgac, N. (2012). Swarm herding using a region holding sliding mode controller. *Journal of Vibration and Control, 18*(7), 1056–1066.

Balch, T., & Arkin, R. C. (1998). Behavior-based formation control for multirobot teams. *IEEE Transactions on Robotics and Automation, 14*(6), 926–939.

Beard, R. W., Lawton, J., & Hadaegh, F. Y. (2001). A coordination architecture for spacecraft formation control. *IEEE Transactions on Control Systems Technology, 9*(6), 777–790.

Bhattacharya, S., Kim, S., Heidarsson, H., Sukhatme, G. S., & Kumar, V. (2015). A topological approach to using cables to separate and manipulate sets of objects. *The International Journal of Robotics Research, 34*(6), 799–815.

Butler, Z., Corke, P., Peterson, R., & Rus, D. (2004). Virtual fences for controlling cows. In *International conference on robotics and automation* (Vol. 5, pp. 4429–4436). New York: IEEE.

Cortes, J., Martinez, S., Karatas, T., & Bullo, F. (2004). Coverage control for mobile sensing networks. *IEEE Transactions on Robotics and Automation, 20*(2), 243–255.

Dantam, N. T., Kingston, Z. K., Chaudhuri, S., & Kavraki, L. E. (2018). An incremental constraint-based framework for task and motion planning. *The International Journal of Robotics Research, 37*(10), 1134–1151.

Durrant-Whyte, H., Roy, N., & Abbeel, P. (2012). Controlling wild bodies using linear temporal logic. *Robotics: Science and Systems, 7,* 17–24.

Edelsbrunner, H., & Harer, J. (2010). *Computational topology: An introduction*. New York: American Mathematical Society.

Egerstedt, M., & Hu, X. (2001a). Formation constrained multi-agent control. *IEEE Transactions on Robotics and Automation, 17*(6), 947–951.

Egerstedt, M., & Hu, X. (2001b). Formation constrained multi-agent control. *IEEE transactions on Robotics and Automation, 17*(6), 947–951.

Elamvazhuthi, K., Kakish, Z., Shirsat, A., & Berman, S. (2020). Controllability and stabilization for herding a robotic swarm using a leader: A mean-field approach. *IEEE Transactions on Robotics, 2020,* 1–15.

Ferrari-Trecate, G., Egerstedt, M., Buffa, A., & Ji, M. (2006). Laplacian sheep: A hybrid, stop-go policy for leader-based containment control. In *International workshop on hybrid systems: Computation and control* (pp. 212–226). London: Springer.

Fine, B. T., & Shell, D. A. (2013). Eliciting collective behaviors through automatically generated environments. In *2013 IEEE/RSJ international conference on intelligent robots and systems* (pp. 3303–3308).

Fink, J., Hsieh, M. A., & Kumar, V. (2008). Multi-robot manipulation via caging in environments with obstacles. In *International conference on robotics and automation* (pp. 1471–1476). New York: IEEE.

Garrell, A., Sanfeliu, A., & Moreno-Noguer, F. (2009). Discrete time motion model for guiding people in urban areas using multiple robots. In *International conference on intelligent robots and systems*. New York: IEEE.

Garrido, S., Moreno, L., & Lima, P. U. (2011). Robot formation motion planning using fast marching. *Robotics and Autonomous Systems, 59*(9), 675–683.

Gazi, V., & Passino, K. M. (2004). A class of attractions/repulsion functions for stable swarm aggregations. *International Journal of Control, 77*(18), 1567–1579.

LaValle, S. M., & Kuffner, J. J., Jr. (2001). Randomized kinodynamic planning. *The International Journal of Robotics Research, 20*(5), 378–400.

Lee, W., & Kim, D. (2017). Autonomous shepherding behaviors of multiple target steering robots. *Sensors (Basel, Switzerland), 17,* 2729.

Lien, J. M., Rodriguez, S., Malric, J. P., & Amato, N. M. (2005). Shepherding behaviors with multiple shepherds. In *International conference on robotics and automation* (pp. 3402–3407).

Lu, Z. (2010). *Cooperative optimal path planning for herding problems*. Ph.D. thesis, Texas A&M University.

Mahler, J., Pokorny, F. T., McCarthy, Z., van der Stappen, A. F., & Goldberg, K. (2016a). Energy-bounded caging: Formal definition and 2-D energy lower bound algorithm based on weighted alpha shapes. *IEEE Robotics and Automation Letters, 1*(1), 508–515.

Mahler, J., Pokorny, F. T., Niyaz, S., & Goldberg, K. (2016b). Synthesis of energy-bounded planar caging grasps using persistent homology. In *Workshop on the algorithmic foundations of robotics*.

Makita, S., & Wan, W. (2017). A survey of robotic caging and its applications. *Advanced Robotics, 31*(19–20), 1071–1085.

Pereira, G. A. S., Das, A. K., Kumar, R. V., & Campos, M. F. M. (2003). *Decentralized motion planning for multiple robots subject to sensing and communication constraints*. Departmental Papers (MEAM) University of Pennsylvania.

Pereira, G. A. S., Campos, M. F. M., & Kumar, V. (2004). Decentralized algorithms for multi-robot manipulation via caging. *The International Journal of Robotics Research, 23*(7–8), 783–795.

Pierson, A., & Schwager, M. (2018). Controlling noncooperative herds with robotic herders. *IEEE Transactions on Robotics, 34*(2), 517–525.

Rodriguez, A., Mason, M. T., & Ferry, S. (2012). From caging to grasping. *The International Journal of Robotics Research, 31*(7), 886–900.

Schultz, A., Grefenstette, J. J., & Adams, W. (1996). Robo-shepherd: Learning complex robotic behaviors. In *Robotics and manufacturing: Recent trends in research and applications* (Vol. 6, pp. 763–768). New York: ASME Press.

Shedied, S. A. (2002). *Optimal control for a two player dynamic pursuit evasion game: The herding problem*. Ph.D. thesis, Virginia Tech.

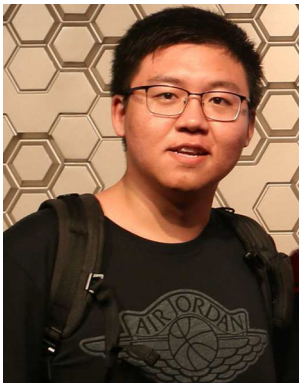
Strömbom, D., Mann, R. P., Wilson, A. M., Hailes, S., Morton, A. J., Sumpter, D., & King, A. J. (2014). Solving the shepherding problem: Heuristics for herding autonomous, interacting agents. *Journal of The Royal Society Interface, 11*(100), 20140719.

Tanner, H. G. (2004). ISS properties of nonholonomic vehicles. *Systems and Control Letters, 53*(3–4), 229–235.

Tanner, H. G., Jadbabaie, A., & Pappas, G. J. (2007). Flocking in fixed and switching networks. *IEEE Transactions on Automatic control, 52*(5), 863–868.

Varava, A., Hang, K., Kragic, D., & Pokorny, F. T. (2017). Herding by caging: A topological approach towards guiding moving agents via mobile robots. In *Proceedings of robotics: science and systems*.

Vaughan, R., Sumpter, N., Henderson, J., Frost, A., & Cameron, s. (2000). Experiments in automatic flock control. *Robotics and Autonomous Systems, 31*(1–2), 109–117.



Haoran Song received the B.S. degree in engineering from Harbin Institute of Technology, China in 2016. He is currently a Ph.D. candidate in the Robotics Institute, Department of Mechanical and Aerospace Engineering, Hong Kong University of Science and Technology, Hong Kong SAR, China. He was a visiting scholar at the Robotics, Perception and Learning Lab, KTH Royal Institute of Technology, Stockholm, Sweden from March to September 2018. His research interests include motion planning and deep learning for robotics.

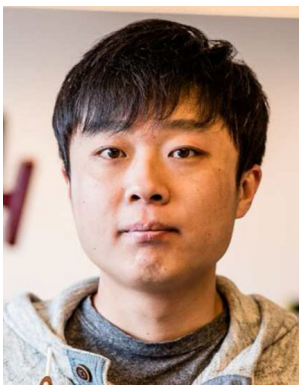
Anastasiia Varava received her M.Sc. degree in Computer Science from the University of Nice Sophia Antipolis, France, in 2014. She received her Ph.D. degree from KTH Royal Institute of Technology, Stockholm, Sweden in 2019. She is interested in geometric and topological applications to Robotics and Machine Learning.

Oleksandr Kravchenko received his M.Sc. degree from Kharkiv National University, Ukraine, in 2015. He received his Ph.D. degree from KTH Royal Institute of Technology, Stockholm, Sweden in 2020. His research interests include geometric and graph representations in robotics and machine learning applications for natural sciences.

Danica Kragic received the M.Sc. degree in Mechanical Engineering from Technical University of Rijeka, Rijeka, Croatia, in 1995 and the Ph.D. degree in computer science from Royal Institute of Technology, KTH, Stockholm, Sweden, in 2001. She is currently a Professor in the School of Electrical Engineering and Computer Science, Royal Institute of Technology, KTH. She had been a Visiting Researcher with Columbia University, Johns Hopkins University, and INRIA Rennes. She is the Director of the Centre for Autonomous Systems. Her research interests include the areas of robotics, computer vision, and machine learning. Dr. Kragic received the 2007 IEEE Robotics and Automation Society Early Academic Career Award. She is a member of the Royal Swedish Academy of Sciences and Young Academy of Sweden. She holds an Honorary Doctorate from Lappeenranta University of Technology, Lappeenranta, Finland. She chaired the IEEE RAS Technical Committee on Computer and Robot Vision and served as an IEEE RAS AdCom member. In 2012, she received an ERC Starting Grant.

Michael Yu Wang obtained his undergraduate degree from Xian Jiaotong University in China, his M.S. degree from Pennsylvania State University, and his Ph.D. from the Carnegie Mellon University. He is currently a Professor in the Department of Mechanical and Aerospace Engineering, Hong Kong University of Science and Technology, and he is the director of HKUST Robotics Institute. Before joining HKUST in 2015, he served on the engineering faculty at University of Maryland, Chinese University of Hong Kong, and National University of Singapore. Professor Wang has held other positions worldwide, including visiting scholar at Stanford University, A*STAR OAP Fellow in Nanyang Technological University in Singapore, Guest Professor in Shanghai Jiaotong University and Haerbin Institute of Technology, Chang Jiang Chair Professor at Huazhong University of Science and Technology, Thousand Talents Professor at Xian Jiaotong University in China, Eminent Visiting Professor at Universiti Brunei Darussalam, and Distinguished Visiting Scholar of University of Technology Sydney. He is the Editor-in-Chief of IEEE Trans. on Automation Science and Engineering and an editor for Computer-Aided Design, Structural and Multidisciplinary Optimization, and Frontiers of Mechanical Engineering. His research focuses primarily on robotics, autonomous systems, manufacturing automation, topology optimization, and additive manufacturing, with over 300 technical publications in these areas.

Florian T. Pokorny received the B.Sc. degree in mathematics from the University of Edinburgh, Edinburgh, UK, in 2005 and completed Part III of the Mathematical Tripos with the University of Cambridge, Cambridge, UK, in 2006. He then received the Ph.D. degree in mathematics from the University of Edinburgh in 2011. In 2011, he joined the KTH Royal Institute of Technology, Stockholm, Sweden, as a Post-Doctoral Researcher. From 05/2015 to 04/2016, he conducted Post-Doctoral Research with the AMPLab and the Berkeley Automation Science and Engineering Laboratory, University of California at Berkeley, Berkeley, CA, USA, under the supervision of Prof. K. Goldberg. He is currently an Associate Professor with the Robotics, Perception and Learning Lab, KTH Royal Institute of Technology. His research interests include robotic manipulation, machine learning, and topological data analysis.



Kaiyu Hang received his B.S. degree in Information Engineering from Xi'an Jiaotong University, Xi'an, China, in 2010, and M.Sc. degree in Communication Systems and Ph.D. in Computer Science, specialized in Robotics and Computer Vision, from KTH Royal Institute of Technology, Stockholm, Sweden, in 2012 and 2016, respectively. He is currently a postdoctoral associate at the GRAB Lab, Yale University, CT, USA. His research interests include representations and optimization for robotic manipulation, motion planning, adaptive grasping and in-hand manipulation, underactuated robotic hands, dual-arm manipulation, and mobile manipulation.