Distributed Dynamic Density Coverage for Human-Swarm Interactions

Yancy Diaz-Mercado, Sung G. Lee and Magnus Egerstedt

***Abstract*— This paper presents two approaches to externally influence a team of robots by means of time-varying density functions. These density functions represent rough references for where the robots should be located. Recently developed continuous-time algorithms move the robots so as to provide optimal coverage of a given the time-varying density functions. This makes it possible for a human operator to abstract away the number of robots and focus on the general behavior of the team of robots as a whole. Using a distributed approximation to this algorithm whereby the robots only need to access information from adjacent robots allows these algorithms to scale well with the number of robots. Simulations and robotic experiments show that the desired behaviors are achieved.**

I. INTRODUCTION

With the advent of new, smaller and more powerful technology, smaller and cheaper robots have made the study of multi-robot systems gain popularity in the academic community. In particular, there has been a lot of stride in the area of human-swarm interaction (HSI), where a single human operator is able to influence large teams of robots [1]–[3]. Using the work in [4], we propose two methods for HSI with which to influence teams of robots. These are in no way exhaustive, nor are they the optimal way of influencing multi-robot systems, they merely provide a foundation for future development and refinement of distributed, dynamic density coverage for HSI ideas.

The organization of this paper is as follows. Section II provides a brief description of recently developed math and update laws that serve as the backbone for the HSI methods presented. Section III begins discussion on how to use these tools to influence multi-robot teams. In this section we also present two HSI methods, one is tailored for obtaining desired geometric configurations, the other for quick manipulation of the swarm as a whole. These methods are tested through simulation and robotic implementation.

This work was supported by a grant from the US Air Force Office of Scientific Research.

The authors are with the Department of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, Georgia, USA yancy.diaz@gatech.edu, slee656@gatech.edu, magnus@gatech.edu

II. MULTI-ROBOT CONTROL USING TIME-VARYING DENSITY FUNCTIONS

The proposed human-swarm interfaces discussed in subsequent sections rely on recently developed control laws that achieve optimal coverage of time-varying densities in a distributed fashion [4], which extends on the work of [5]–[10]. For the sake of completeness, in this section we recall some of the formulations described in [4] and refer the readers to this for a more wholesome discussion on these control laws. We will however elaborate on some of the more subtle difficulties not discussed in [4] to enrich its content.

A. Time-Varying Density Functions: Centralized Case

In order to discuss optimal coverage, we must first associate the configuration of the team of robots to a cost on how well the team is covering the area of interest. We denote $D \subset \mathbb{R}^2$ to be the two-dimensional convex domain representing the area of interest. Within the area of interest, we wish to deploy a team of n robots with positions $p_i \in D$, $i = 1,2,\ldots,n$, and influence this team of robots to achieve particular configurations. For convenience, we can collect the position of the agents into a single vector $p = \left[p_1^T,\ldots,p_n^T\right]^T$. The configurations are specified by assigning some relative importance to the points in D at a given time. We let $\phi : D \times [0,\infty) \to (0,\infty)$ be a bounded density function, C^1 in both arguments. The relative importance of a point $q \in D$ at time t is then captured by $\phi(q,t)$. Since the performance of a large class of sensors deteriorates with a rate proportional to the square of the distance [11], [12], we consider the following so-called locational cost

$$H(p,t) = \sum_{i=1}^{n} \int_{V_i(p)} \|q - p_i\|^2 \phi(q,t)\, \mathrm{d}q \qquad (1)$$

where the domain D has been partitioned into regions of dominance (e.g., [7]), letting each agent in charge of covering only their region. In particular, we utilize a Voronoi tessellation since it has been shown to minimize the locational cost for given p, [6]. The Voronoi cell for agent i would be given by

$$V_i(p) = \{q \in D \mid \|q - p_i\| \leq \left\|q - p_j\right\|, i \neq j\}. \qquad (2)$$

In [13], [14] it was shown that

$$\frac{\partial H}{\partial p_i} = \int_{V_i} -2(q-p_i)^T \phi(q,t) dq = 2m_i(p_i - c_i)^T, \quad (3)$$

where one can define the mass m_i and center of mass c_i of the i^{th} Voronoi cell, V_i, as

$$m_i(p,t) = \int_{V_i(p)} \phi(q,t) dq, \quad (4)$$

$$c_i(p,t) = \int_{V_i(p)} q\phi(q,t) dq \Big/ m_i \quad (5)$$

since $\phi > 0$. From (3), it can be seen that a critical point of (1) is

$$p_i(t) = c_i(p,t), \quad i = 1, \cdots, n, \quad (6)$$

and so a minimizer to (1) is necessarily in this form, [15]. Moreover, when (6) is satisfied, p is a so-called centroidal Voronoi tessellation (CVT).

Let $c = \left[c_1^T, \ldots, c_n^T\right]^T$. Using the above notation, we present the following result, originally stated in [4]

Theorem 2.1: Let $p(t_0) = c(p(t_0), t_0)$. If

$$\dot{p} = \left(I - \frac{\partial c}{\partial p}\right)^{-1} \frac{\partial c}{\partial t}, \qquad t \geq t_0$$

then

$$\|p(t) - c(p(t), t)\| = 0, \qquad t \geq t_0$$

as long as the inverse $(I - \partial c/\partial p)^{-1}$ is well-defined.

Proof: Assume the agents begin from a CVT configuration, i.e., $p(t_0) = c(p(t_0), t_0)$. We need to ensure that

$$\frac{\mathrm{d}}{\mathrm{d}t}(p(t) - c(p(t), t)) = 0, \qquad \forall t \geq t_0.$$

But this implies that $\dot{p} = \dot{c} = \frac{\partial c}{\partial p}\dot{p} + \frac{\partial c}{\partial t}$, which can be rearranged into the form

$$\dot{p} = \left(I - \frac{\partial c}{\partial p}\right)^{-1} \frac{\partial c}{\partial t}$$

as long as the inverse is well-defined. ■

However, in order for this theorem to be meaningful, we require that the agents first achieve at a CVT configuration which could then be maintained. Moreover, this control law assumes the agents will be able to instantaneously accelerate to the required velocities, which may not be possible in practice. In order compensate for saturation, modeling errors and deviations from the CVT, in [4] a proportional term is introduced that forces the agents into a CVT. This update law was called the *TVD-C* which stands for Time-Varying Densities, Centralized case:

$$\dot{p} = \left(I - \frac{\partial c}{\partial p}\right)^{-1} \left(-\kappa(p - c) + \frac{\partial c}{\partial t}\right) \quad (7)$$

where $\kappa > 0$ is a proportional gain. It is noteworthy that this proportional term influences the team of robots to move towards a (scaled) gradient descent direction to achieve a CVT configuration, and that once a CVT is achieved the proportional term does not contribute to the update law and theorem 2.1.

Two subtle difficulties remain with update law (7). The first is the computation of the deceivingly innocent looking terms $\partial c/\partial p$ and $\partial c/\partial t$. Recall that by combining equations (4) and (5), we have that

$$c_i(p,t) = \frac{\int_{V_i(p)} q\,\phi(q,t)\,\mathrm{d}t}{\int_{V_i(p)} \phi(q,t)\,\mathrm{d}t},$$

which depends on p in the boundary of the area over which the two integrals are taken.

In order to compute these partials, we'll first need to make use of Leibniz rule, e.g., [16].

Lemma 2.2: Let $\Omega(p)$ be a region that is a smooth function of p such that the unit outward normal vector n is uniquely defined almost everywhere on $\partial\Omega$, which is the boundary of Ω. Let

$$F = \int_{\Omega(p)} f(q) dq.$$

Then

$$\frac{\partial F}{\partial p} = \int_{\partial\Omega(p)} f(q)\hat{q} \cdot n(q) dq$$

where $\hat{q}$ is the derivative of the points on $\partial\Omega$ with respect to p.

In [16], it was investigated how Voronoi cells changed as functions of p_i. In fact, it was shown in [16] that for any point $q \in \partial V_{i,j}$ (the boundary between adjacent cells V_i and V_j),

$$\frac{\partial q}{\partial p_j^{(b)}} \cdot (p_j - p_i) = \frac{1}{2} e_b \cdot (p_j - p_i) - e_b \cdot \left(q - \frac{p_i + p_j}{2}\right),$$

$$\frac{\partial q}{\partial p_i^{(b)}} \cdot (p_j - p_i) = \frac{1}{2} e_b \cdot (p_j - p_i) + e_b \cdot \left(q - \frac{p_i + p_j}{2}\right),$$

where $p_j^{(b)}$ denotes the b^{th} component of the vector p_j and e_b is the b^{th} elementary unit vector. Note that in this paper, $b = 1, 2$ since we are considering the case $D \subset \mathbb{R}^2$ only.

Substituting this into Leibniz rule, we obtain

$$\frac{\partial c_i^{(a)}}{\partial p_j^{(b)}} = \left(\int_{\partial V_{i,j}} \phi q^{(a)} \frac{p_j^{(b)} - q^{(b)}}{\|p_j - p_i\|} dq\right) \Big/ m_i$$

$$- \left(\int_{\partial V_{i,j}} \phi \frac{p_j^{(b)} - q^{(b)}}{\|p_j - p_i\|} dq\right) \left(\int_{V_i(P)} \phi q^{(a)} dq\right) \Big/ m_i^2 \quad (8)$$

where $a = 1,2$, $b = 1,2$ and where $i \neq j$. When $i = j$ we must consider the contribution from all neighbors

$$\frac{\partial c_i^{(a)}}{\partial p_i^{(b)}} = \sum_{k \in N_{V_i}} \left[\left(\int_{\partial V_{i,k}} \phi q^{(a)} \frac{q^{(b)} - p_i^{(b)}}{\|p_k - p_i\|} dq \right) \Big/ m_i \right.$$
$$\left. - \left(\int_{\partial V_{i,k}} \phi \frac{q^{(b)} - p_i^{(b)}}{\|p_k - p_i\|} dq \right) \left(\int_{V_i(P)} \phi q^{(a)} dq \right) \Big/ m_i^2 \right] \quad (9)$$

which gives us all we need to compute $\partial c/\partial p$. It is noteworthy that given a continuously differentiable density function ϕ, computing $\partial c/\partial p$ at any given time t becomes an exercise in finding line and area integrals. In implementation, it suffices to use numerical approximations to obtain these integrals (e.g., Riemann sums and Gaussian quadrature rule).

One more partial derivative is required for update law (7), namely $\partial c/\partial t$. Another application of Leibniz rule results in

$$\frac{\partial c_i}{\partial t} = \frac{m_i \int_{V_i} q \frac{\partial \phi}{\partial t}(q,t) dq - \frac{\partial m_i}{\partial t} \int_{V_i} q \phi(q,t) dq}{m_i^2} \quad (10)$$

with $\frac{\partial m_i}{\partial t} = \int_{V_i(p)} \frac{\partial \phi}{\partial t}(q,t) dq$ and $\frac{\partial c}{\partial t}^T = [\frac{\partial c_1}{\partial t}^T \ \cdots \ \frac{\partial c_n}{\partial t}^T]$.

Note that in implementation, this is again a matter of numerically computing integrals. However, unlike with $\partial c/\partial p$, we require knowledge of $\partial \phi/\partial t$. If ϕ is not provided analytically in t, then one could:

i Utilize a finite difference scheme to approximate $\partial \phi/\partial t$. This could however give rise to difficulties with noisy measurements. This approach is utilized in the implementation found in section III-B.

ii Alternatively, the user could provide the "shape" of the density with a continuously differentiable function $\phi(q,t_0)$ defined over D, and define its time evolution directly via a continuous function $\partial \phi/\partial t$ such that $\phi(q,t) = \int_{t_0}^{t} \frac{\partial \phi}{\partial t}(q,\tau)\, \mathrm{d}\tau$. This is the approach used in the implementation found in section III-A.

The second subtle difficulty with update law (7) is ensuring that the inverse $(I - \partial c/\partial p)^{-1}$ is well defined, which is in general hard to do. In [16] it was shown that in the time-invariant case, the inverse is well-defined as long as $\phi(p)$ is a log-concave function of p. We would also need ϕ to be continuously differentiable in both arguments, so these two conditions are enough to ensure that the inverse exists. Since the motivation for this work is to have a human operator influence the team of robots by generating these densities functions, the former constraint is quite an unsatisfying one, for it would greatly reduce the types of density functions allowable. Moreover, the computation of this $2n \times 2n$ matrix inversion does not scale well with increase in number of agents, and requires information on every agent which makes it a centralized scheme. Fortunately, it is possible to alleviate these concerns by performing a distributed approximation of update law (7).

B. Distributed Approximations

In [4], a family of distributed approximations to update law (7) were presented which trades-off having to find the computationally costly matrix inverse $(I - \partial c/\partial p)^{-1}$ with imposing network connectivity constraints on the team of robots. As a consequence, we can discard the use of the matrix inverse, which could be ill-defined under certain conditions, with a well-defined series approximation. These distributed update laws lie at the heart of the proposed human-swarm interfaces, so a brief treatment of their derivation is included for the sake of completeness. The required approximation can be found by using the Neumann series, e.g., [17].

Lemma 2.3 (Neumann series): Let A be a square matrix. If $\lim_{k\to\infty} A^k = 0$, then $I - A$ is invertible and

$$(I - A)^{-1} = I + A + A^2 + A^3 + \ldots.$$

Moreover, for a $m \times m$ square matrix A, $\lim_{k\to\infty} A^k = 0$ if and only if $|\lambda_i| < 1$ for all $i = 1,2,\cdots,m$, where λ_i are the eigenvalues of A. As such, let λ_{max} denote the eigenvalue with the largest magnitude of the matrix $\partial c/\partial p$. Using the Neumann series, we can express $(I - \partial c/\partial p)^{-1}$ as

$$\left(I - \frac{\partial c}{\partial p}\right)^{-1} = I + \frac{\partial c}{\partial p} + \left(\frac{\partial c}{\partial p}\right)^2 + \ldots \quad (11)$$

as long as $|\lambda_{max}| < 1$.

Our goal will be to truncate this series to obtain the well-defined approximation to the matrix inverse, but then the question arises: how many terms should be kept? The answer lies in the sparsity structure of $\partial c/\partial p$.

Given a Voronoi partition of the area of interest, we denote the boundary between the two cells V_i and V_j by ∂V_{ij}. Since we are only considering the planar case, there are three possibilities for ∂V_{ij}:

i ∂V_{ij} is empty, meaning that cells V_i and V_j do not intersect.

ii ∂V_{ij} consist of a single point, meaning that cells V_i and V_j share a single vertex.

iii ∂V_{ij} is a line, meaning that cells V_i and V_j share a face.

We will denote N_{V_i} to be set of indexes pertaining to the agents whose Voronoi cells V_j share a face with agent i's Voronoi cell V_i.

Lemma 2.4: $j \notin N_{V_i} \implies \dfrac{\partial c_i}{\partial p_j} = 0.$

Proof: For the first two cases, i.e., ∂V_{ij} is either empty or consists of a singleton, from (8) and (9) we see that the integrals over ∂V_{ij} would be zero. Note that for these two cases, this will be true for all four elements in $\partial c_i/\partial p_j$. Since these two cases correspond to agents i and j not sharing a face, we conclude that $j \notin N_{V_i}$ implies that $\partial c_i/\partial p_j = 0$. ■

This lemma tells us that $\partial c/\partial p$ actually encodes adjacency information of the graph induced by the Voronoi tessellation. This induced graph is known as the *Delaunay graph*. To obtain a distributed update law, we must insist that the update for $\dot{p}_i$ depends only on information from itself (p_i and $\phi(q,t)$ for $q \in V_i$) and information on neighboring agents (p_j and $\phi(q,t)$ for $q \in V_j$, for all $j \in N_{V_i}$). To this end, we truncate the Neumann series in (11) after just two entries, i.e., $\left(I - \partial c/\partial p\right)^{-1} \approx I + \partial c/\partial p$. By modifying update law (7) with this approximation, we obtain the update law

$$\dot{p} = \left(I + \frac{\partial c}{\partial p}\right)\left(-\kappa(p-c) + \frac{\partial c}{\partial t}\right),$$

which at the individual robot level results in the update law called *TVD-D*$_1$ for Time-Varying Densities, Distributed case with 1-hop adjacency information:

$$\dot{p}_i = \frac{\partial c_i}{\partial t} - \kappa(p_i - c_i) + \sum_{j \in N_{V_i}} \frac{\partial c_i}{\partial p_j}\left(\frac{\partial c_j}{\partial t} - \kappa(p_j - c_j)\right). \tag{12}$$

It should be noted that (12) is always well-defined (as long as ϕ is continuously differentiable). In other words, even if the Neumann series is not convergent or if the inverse does not exist, the entries in (12) are well-defined. In fact, it turns out that during the robotic experiment, even in cases where $|\lambda_{max}| > 1$, the robots consistently evolve in a manner that achieves coverage. In [4], a comparison is made between both the centralized and decentralized approaches shown here to other available techniques for coverage of time-varying densities — we refer the readers to it for a discussion on how they compare.

III. Densities for Human-Swarm Interfaces

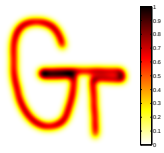
The fact that only adjacency information is required in update law (12) means that a single operator could potentially influence arbitrarily large number of agents. Due to the scalability of the algorithm, and the level of abstraction that the generation of density functions offers (in that it does not care for the number of agents performing the coverage), it is possible to use (12) as a tool for human-swarm interaction (HSI) in order to allows for human operators to influence teams of robots by generating density functions and having these robots perform coverage. However, how should the human operator generate these densities? In this section we offer two different means of generating density functions for HSI. The first method allows the user to easily specify the geometric configuration of the swarm and "move" them around, but at an added computational cost. The second method reduces computational cost by using predefined Gaussian functions to allow the human operator to quickly assign importance to the area of interest. It is fast and allows the user to "move" the team of robot easily, but is not as amenable to "shaping" the swarm as the first method.

A. Diffusion of Drawn Geometric Configurations

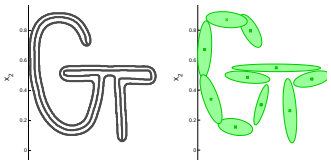
We now present an approach that allows the user to specify the geometric configuration of the swarm. The process consists of allowing the human operator to draw the desired shape in the tablet-like interface. This drawing, taken as a binary image over the area of interest $\Psi : D \to \{0,1\}$, can be made smooth and turned into a continuously differentiable function in the spatial argument by evolving it as a diffusion process. The end result is a smooth, non parametric density function with the pixels determining the density intensities. One could then simply pass the image to the numerical integrating tools to obtain update law for the agents, interpolating between pixel values as needed. Fig. 1a illustrates an example of the smoothed drawing that represents the desired density.

However, in order to reduce the amount of information needed to be communicated to the agents, it is possible to use the level sets of the resulting image to come up with a Gaussian Mixture Model (GMM), which would collapse the dimensionality from the pixel count (which could be significant) to k centroids and covariances, k being the amount of Gaussian functions desired to approximate the non parametric density, thus reducing significantly the amount of information required to represent the density.

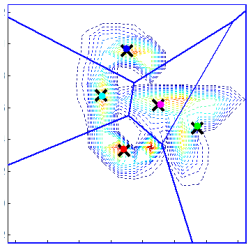
In [18], a process to obtain a GMM from a data set with redundancies over the data is presented. This process can be utilized to obtain a parametric approximation of the desired density. In order to generate the required redundant data sets, sample points can be selected from the contour level sets of the desired density. The points in this data set are then grouped into k clusters, where the design parameter k is the number of Gaussian models used in the GMM. A larger k can be used for a finer approximation of the desired density whereas a smaller k can be used for coarser approximation. The data from each cluster is used to determine the parameters for each



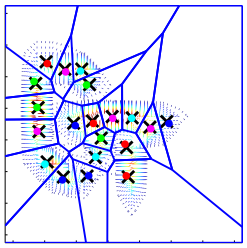
(a) Smooth Nonparametric Density



(b) Level Set Sample Data & GMM Parametric Density Estimate



(c) Coverage by 5 agents.



(d) Coverage by 20 agents.

Fig. 1: Distributed coverage simulation of a time-varying, user provided density.

of the Gaussian models. Fig. 1b illustrates the data set obtained from the original drawing with the redundant data points obtained from the level set contours of the desired density function. Fig. 1b also illustrates the GMM found to approximate the density with the use of nine Gaussian functions.

The GMM illustrated in Fig. 1b was used as an input the $TVD-D_1$. The density was made time-varying by making every Gaussian function's centroid follow a circular orbit, i.e., the density was in the general form

$$\phi(q,t) = \sum_{i=1}^{n} \frac{\alpha_i}{2\pi\sqrt{\det(\Sigma_i)}} e^{-\frac{1}{2}(\mu_i - v(t) - q)^T \Sigma_i^{-1} (\mu_i - v(t) - q)}$$

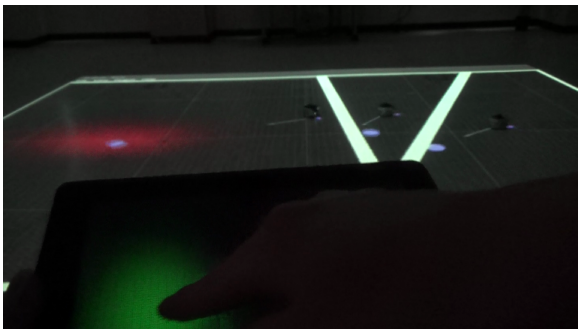
where the parameters α_i are such that $\sum_{i=1}^{n} \alpha_i = 1$, μ_i and Σ_i correspond to the centroid and covariance for the i^{th} Gaussian function in the GMM of n Gaussian functions, and $v(t) = (r\sin(bt), r\cos(bt))^T$ with r and b being fixed parameters used to determine the time evolution of the circular orbit. Fig. 1 illustrates a team of five and 20 robots performing coverage of the GMM with $n = 9$, $r = 0.2$ and $b = \pi/4$. The contours of the density are illustrated with dotted lines, solid lines are used as the boundaries of the Voronoi cells, the crosses correspond to the centroid of each Voronoi cell and the circles represent the agents performing the coverage.

B. Control of Gaussian Functions

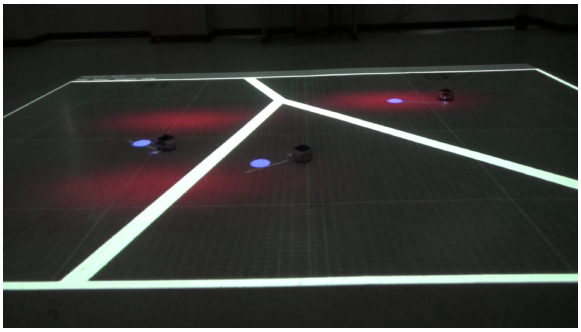
The previous approach allows an user to generate a density function that captures the important areas to be covered by simply drawing over the domain. In order to reduce the amount of information required to describe the generated non parametric density, this was approximated by a Gaussian Mixture Model (GMM) with k Gaussian functions. However, if the human operator is less interested in the geometric shape the swarm takes and more interested in actively manipulating the swarm by dragging it as a whole, splitting and merging it, then GMM concept from the previous method can be modified by discarding the a priori weights and fixing the general "shape" of the Gaussian functions. This greatly reduces computation and allows the user to provide quick references on where the agents should concentrate.

The method involves allowing the human operator to "tap" on a tablet the spots where the agents should concentrate. Influence on the team of robots is achieved by performing coverage of the density function made of Gaussian functions in the form $\phi(q,t) = \frac{1}{M}\sum_{\ell=1}^{M} e^{-\frac{1}{2}(q-\mu_\ell)^T \Sigma^{-1}(q-\mu_\ell)}$, where Σ is a positive definite two dimensional matrix that determines the "shape" for the the Gaussian functions, M is the number of fingers the user is tapping with and μ_ℓ is the location of the fingers in the plane and will also correspond to the centroid of the Gaussian functions, $\ell = 1,\ldots,M$.

The control of Gaussian functions method was implemented in a multi-robot system, where the human operator used an iPad to input the centroid of the Gaussian functions. These were transmitted over WiFi and UDP to an Ubuntu (version 11.04) computer with an Intel dual core CPU 2.13GHz and 4GB of memory, running ROS (Robot Operating System, version Diamondback). This computer also received state information from ten OptiTrack S250e motion capture cameras that were used to provide position and orientation data for the robots. The state information of an agent and its neighbors was used to compute the Voronoi tessellation. For every agent, only it and its neighbors' state and density information in their respective Voronoi cells were used to compute the control. Line integrals were approximated by Riemann sums, whereas Gaussian quadrature rule was used for area integrals. A finite difference scheme was used to approximate $\partial\phi/\partial t$ based on stored samples of the density function. Even though a central computer is used for the computation, the control is computed for every agent only using adjacency information, and then it is transmitted to the pertinent robot via WiFi and UDP. The robotic platforms used for the experiments were Khepera III robots from K-team – differential drive wheeled robots equipped with a wireless card for communication over a wireless router. The rviz package in ROS was used for visualizations, such as the position and the orientation of the robots, the density function, and the Voronoi partitions. The visualization was overlapped with the real physical environment to give a real-time



(a) Using only one finger.



(b) Using three fingers.

Fig. 2: Multi-robot implementation of the control of Gaussian functions method for HSI.

visual representation. This is shown in Fig. 2, where the solid lines represent the boundary of the Voronoi cells and the small circles represent their centroid. The arrows emerging from the robots point in the direction of motion as determined by the update law.

As the Khepera III mobile robots are differential-drive robots, they can be modeled as unicycles, i.e., $\dot{x}_i = v_i \cos\theta_i$, $\dot{y}_i = v_i \sin\theta_i$, and $\dot{\theta}_i = \omega_i$, where (x_i, y_i) is the position of robot i on the plane, θ_i its heading, and v_i, ω_i are the translational and angular velocities. In contrast to this, the coverage algorithm provides desired motions in terms of $\dot{p}_i$ and we map these onto v_i, ω_i through $v_i = \|\dot{p}_i\|$, and $\omega_i = [-\sin\theta_i, \cos\theta_i] \cdot \dot{p}_i / \|\dot{p}_i\|$.

IV. Conclusions

This paper presented two preliminary approaches to HSI using the update laws presented in [4]. One approach generates a C^1 density functions that allows the user to easily describe the desired geometric configuration the swarm should take by drawing the general shape on a tablet-like interface. The second approach allows the human operator to quickly manipulate the position of the swarm when the geometric configuration is not the main objective. The user only needs to "tap" on a tablet-like interface to provide a density for the robots, without the computational burden of generating the density function the previous method incurs.

References

[1] S. Bashyal and G. Venayagamoorthy, "Human swarm interaction for radiation source search and localization," in *Swarm Intelligence Symposium, 2008. SIS 2008. IEEE*, Sept 2008, pp. 1–8.

[2] M. Diana, J.-P. de la Croix, and M. Egerstedt, "Deformable-medium affordances for interacting with multi-robot systems," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, Nov 2013, pp. 5252–5257.

[3] E. Schoof, A. Chapman, and M. Mesbahi, "Bearing-compass formation control: A human-swarm interaction perspective," in *American Control Conference (ACC), 2014*, June 2014, pp. 3881–3886.

[4] S. G. Lee, Y. Diaz-Mercado, and M. Egerstedt, "Multi-Robot Control Using Time-Varying Density Functions," *IEEE Transactions on Robotics*, To Appear.

[5] S. Lloyd, "Least squares quantization in PCM," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, Sept. 2006.

[6] J. Cortes, S. Martinez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks: Variations on a theme," in *Mediterranean Conference on Control and Automation*, Lisbon, Portugal, July 2002, Electronic Proceedings.

[7] ——, "Coverage control for mobile sensing networks," *IEEE Transactions on Robotics and Automation*, vol. 20, no. 2, pp. 243–255, Apr. 2004.

[8] A. Ghosh and S. K. Das, "Coverage and connectivity issues in wireless sensor networks: A survey," *Pervasive and Mobile Computing*, vol. 4, no. 3, pp. 303–334, 2008.

[9] A. Ghaffarkhah, Y. Yan, and Y. Mostofi, "Dynamic coverage of time-varying environments using a mobile robot – A communication-aware perspective," in *GLOBECOM Workshops (GC Wkshps), 2011 IEEE*, 2011, pp. 1297–1302.

[10] Y. Q. Chen, Z. Wang, and J. Liang, "Automatic dynamic flocking in mobile actuator sensor networks by central voronoi tessellations," in *Mechatronics and Automation, 2005 IEEE International Conference*, vol. 3, 2005, pp. 1630–1635 Vol. 3.

[11] S. Meguerdichian, F. Koushanfar, G. Qu, and M. Potkonjak, "Exposure in wireless Ad-Hoc sensor networks," in *Proceedings of the 7th annual international conference on Mobile computing and networking*, ser. MobiCom '01. New York, NY, USA: ACM, 2001, pp. 139–150.

[12] S. Adlakha and M. B. Srivastava, "Critical density thresholds for coverage in wireless sensor networks," in *Wireless Communications and Networking Conference (WCNC)*. IEEE, 2003, pp. 1615–1620.

[13] M. Iri, K. Murota, and T. Ohya, "A fast Voronoi-diagram algorithm with applications to geographical optimization problems," in *System Modelling and Optimization*, ser. Lecture Notes in Control and Information Sciences, P. Thoft-Christensen, Ed. Springer Berlin Heidelberg, 1984, vol. 59, pp. 273–288.

[14] Q. Du, V. Faber, and M. Gunzburger, "Centroidal Voronoi Tessellations: Applications and Algorithms," *SIAM Review*, vol. 41, no. 4, pp. 637–676, Dec. 1999.

[15] Q. Du, M. Emelianenko, and L. Ju, "Convergence of the Lloyd Algorithm for Computing Centroidal Voronoi Tessellations," *SIAM Journal on Numerical Analysis*, vol. 44, no. 1, pp. 102–119, Jan. 2006.

[16] Q. Du and M. Emelianenko, "Acceleration schemes for computing centroidal Voronoi tessellations," *Numerical Linear Algebra with Applications*, vol. 13, no. 2-3, pp. 173–192, 2006.

[17] G. Stewart, *Matrix Algorithms Volume 1: Basic Decompositions*. Society for Industrial and Applied Mathematics, 1998.

[18] S. Calinon, F. Guenter, and A. Billard, "On Learning, Representing and Generalizing a Task in a Humanoid Robot," *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 37, no. 2, pp. 286–298, 2007.