Christopher G. Valicka, Richard A. Rekoske, Dušan M. Stipanović, and Ali E. Abbas

Multiattribute Utility Copulas for Multi-objective Coverage Control

Abstract: This paper presents theoretical and experimental results related to the control and coordination of multirobot systems interested in dynamically covering a compact domain while remaining proximal, so as to promote robust inter-robot communications, and while remaining collision free with respect to each other and static obstacles. A design for a novel, gradient-based controller using nonnegative definite objective functions and an overapproximation to the maximum function is presented. By using a multiattribute utility copula to scalarize the multi-objective control problem, a control law is presented that allows for flexible tuning of the tradeoffs between objectives. This procedure mitigates the controller's dependence on objective function parameters and allows for the straightforward integration of a novel global coverage objective. Simulation and experiments demonstrate the controller's effectiveness in promoting scenarios with collision free trajectories, robust communications, and satisfactory coverage of the entire coverage domain concurrently for a group of differential drive robots.

Keywords: multiobjective control, multiattribute utility copulas, dynamic coverage, differential drive robots

DOI 10.2478/pjbr-2014-0002
Received January 14, 2014; accepted May 18, 2014.

1 Introduction

Unmanned and autonomous systems continue to be an area of fervent research and industry focus as more and more they represent a cost effective manner to accomplish hazardous and complex tasks with reduced human supervision. Through advances in the reliability and cost of computing and communication capabilities, simple and inexpensive yet powerful subsystems can be combined to produce large-scale systems capable of achieving complex goals. By using well defined objectives and constraints, complex systems comprised of many subsystems, often referred to as multi-agent systems, can accomplish complex goals normally impossible for systems comprised of fewer robots. Challenges in control law design for multi-agent systems arise when various objectives or constraints conflict and additional challenges arise when dynamical models are used to describe the robots. Multi-agent systems may be comprised of homogenous or heterogenous subsystems, necessitating additional considerations to accurately encapsulate constraints and objectives. Differences between robots can have considerable influence on accomplishing objectives such as safety verification, reliable communication, and dynamic coverage. The research presented herein aims to address several of these challenges simultaneously through implementation of a novel controller in simulation and on physical systems, validating the considered approach for heterogenous systems modeled as unicycle dynamical systems.

1.1 Background

Methodologies that address the various problems associated with control and coordination of multi-agent systems intending to accomplish multiple objectives represent an extensive and active area of research and development. A variety of objectives for such systems have been stud-
ied such as optimal placement for sensing in [1], optimal placement for target tracking in [2] and dynamic coverage control as in [3]. In [4], a multitude of related objectives are considered for a hierarchical architected multi-robot system including wall-following, obstacle avoidance, formation control, and cooperative mapping though in general they are considered in separate experiments. Commonly, formation and avoidance objectives for multi-robot systems are modeled using potential field methods. In [5], such methods are used for flocking. In [6] and [7] these methods are used for formation control with the later also using potential methods for collision avoidance. To avoid local minima common to potential field formulations for formation and avoidance control, controllers were implemented on car-like robots in [8] and miniature helicopters in [9] using Lyapunov-like functions.

Amongst the large number of tasks or objectives considered in the context of control and coordination of multi-robot systems, several stand out as essential objectives. Collision avoidance is common in multi-robot systems intending to accomplish any additional objective. Many approaches exist for ensuring collisions do not occur within multi-robot systems including the potential field methods mentioned previously, see for example [10], and hybrid approaches which incorporate safety policies such as in [11]. The scalable and computationally efficient method of optimal reciprocal collision avoidance (ORCA) has been applied to holonomic systems in [12]. Utilizing the velocity state of robots, this work has been extended to consider localization uncertainty and the scenario of two robots with different velocities in [13]. The focus herein considers the formulation originally established in [14] which uses a reactive, cooperative avoidance methodology specified in terms of detection and avoidance regions. This formulation specifies sufficient conditions to guarantee cooperating robots will not enter each other’s avoidance region. Perfect localization information is assumed in the work presented herein, though this formulation has been extended to address bounded localization error in [15]. Ultimately, this formulation was chosen as it uses only robot position states and is not dependent on a robot’s dynamic model.

Formation or proximity based constraints are also particularly relevant to multi-robot systems and represent a broad area of research. See, for example, [8, 16] which are examples of leader-follower arrangements. A detailed analysis of a broad class of flocking behaviors is presented in [17]. Proximity-based objective functions herein consider instead a simple form, as presented in [18], with the purpose of modeling communication constraints. This formulation treats the proximity problem from the perspective of each robot’s particular communications constraints, suitable for a heterogenous and distributed communications scheme. This formulation does not enforce a strict distance constraint, enabling a tradeoff between timely exchange of information and accomplishment of other objectives. This is relevant when combined with the dynamic coverage objective considered in this work as the robots can potentially cover distant points in the coverage domain at the cost of multiple robots redundantly covering the same point.

A particular focus within this work is given to the objective of dynamic coverage. A problem related to dynamic coverage is studied in multi-robot patrol problems. A minimum refresh time patrolling problem is studied in [19] where graph traversal strategies are computed a priori in a centralized manner. Various coordination schemes for the multi-robot patrol problem (MRPP) are compared in [20] on an outdoor distributed robot system. Distributed Bayesian strategies are used to address the MRPP problem and validated experimentally in [21] with robust and scalable performance compared to previous work. Alternatively, for coverage control using path planning methodologies see for example [22] and [23]. In general, both MRPP and path planning formulations assume instantaneous satisfactory sensing capability. The dynamic coverage control formulations considered herein discretize the coverage domain into a uniform grid and model robot sensing capabilities as a function of time. Previous work related to dynamic coverage of single integrator systems was considered in [24]. This work, and others, use various methods, including a switching control law, or assumptions such as the increased sensing radius of a leader robot, to achieve global coverage. [25] considers the dynamic coverage problem in which coverage information decays. The definition of dynamic coverage considered herein follows the definition given in [18] and differs from the definitions considered in [1], which studies the deployment of mobile sensors, and in the problem of persistent coverage presented in [26] which uses predefined desired paths.

Many examples of multi-robot testbeds used for validation of the various formulations of these objectives and their combinations exist. The testbeds include vehicles of various sensing, localization, communication, and locomotive capabilities. Coordinated mapping and exploration by iRobot B21 robots with laser range scanners is presented in [27]. A testbed for differential drive robots equipped with cameras and laser range finders is presented in [28] with these sensors enabling many robots to accomplish cooperative manipulation and other tasks. Formation control is studied for a group of car-like robots that perform local-
ization using the common tool of an overhead camera system in [29] and [30]. Similarly, a testbed for quadrotor control algorithms and their effects on the dynamics of multi-robot systems uses an overhead motion tracking system for localization in [31]. Wheeled robots which utilize GPS sensors for localization towards relative formation control are presented in [32]. The testbed of [33] studies different wireless communication topologies, and fidelities, in the context of trajectory tracking by fan-propelled robots. Task allocation for a multi-robot system of car-like robots interested in accomplishing one of object tracking, sentry duty, cleaning, or object monitoring tasks is studied in [34]. To the authors’ knowledge, this is the first physical implementation of the dynamic coverage control originally conceived in [24] on a multi-robot testbed. The experiments presented herein use common sensing, localization, and communication technologies and aim to validate a framework suitable for studying emerging behavior resulting from the combination of multiple objectives.

1.2 Summary of Contributions

By combining the well established approaches of multi-objective optimization and multiattribute decision analysis, this work presents new tools for designing control laws for multi-robot systems tasked with multiple objectives. A novel method for ensuring that robots satisfactorily cover points in the coverage domain even after entering local minima is presented. By making use of a simple search algorithm and a careful definition of a robot’s nearest uncovered point, an effective global coverage scheme is incorporated without the need to design an additional switching control law.

Details and preliminaries of multiattribute utility theory are presented towards the end goal of adapting a tool originally used in multiattribute decision analysis for use in multi-objective control. Formal proofs are given that establish a relationship between general $n$-dimensional copulas and an approximation of the minimum function from below. Systematic and intuitive methods from multiattribute decision analysis are adapted for determining the tradeoffs between the robots’ respective objectives.

Theoretical results from using multiattribute utility copulas as approximations to the maximum function from above are presented. The relationship between goal functions constructed using multiattribute utility copulas and the maximum function is explicitly demonstrated. Formal definitions of goal functions and of accomplishing all objectives are rigorously established by using the comparison principle, differential inequalities and Lyapunov-like analysis. Ultimately, an explicit gradient-like control law is designed to accomplish multiple objectives simultaneously.

Finally, the ability of goal functions designed using multiattribute utility copulas to streamline the selection of attribute tradeoff parameters is demonstrated through their implementation on simulated unicycle robots as well as on multi-robot experiments with varying numbers of robots. The controller’s ability to incorporate additional objectives is demonstrated by the use of the global coverage objective in both simulations and experiments. A series of simulations explore the effects of varying multiattribute utility copula parameters on multi-robot system performance. Both simulations and experiments also demonstrate the ability of the new global coverage scheme to accomplish global coverage and to be implemented in a straightforward manner. Ultimately, the simulations and experiments validate a framework for creating complex yet achievable behavior through the use of well-defined objective functions and multiattribute utility copula based control laws.

The remaining sections of the paper are organized as follows. Section 2 presents preliminaries related to multiattribute utility copulas. Theory related to the properties of copulas and multiattribute utility copulas is presented and a formal method for assigning preferences amongst the multiple objectives is shown. In Section 3, desired behavior and constraints will be formulated in terms of non-negative definite objective functions for multi-robot systems intending to accomplish multiple objectives. Essential objectives will be presented including formulations for collision avoidance, robust communications, dynamic coverage, and global coverage. Presentation of the considered objectives concludes with a discussion of control law design using goal functions constructed as approximations of the maximum function. A precise definition is given which outlines the relationship between all objectives being satisfied and a goal being accomplished. Sections 4 and 5 present simulation and experimental results from implementations of multiattribute utility copula based control laws on multi-robot systems. Systems defined using unicycle dynamic models are simulated and experiments are conducted using car-like robots. Section 6 presents conclusions of the research conducted to date as well as directions and areas for future research.
2 Properties of Copulas and Multiattribute Utility Functions

The purpose of this section is to review relevant details of multiattribute utility function theory with salient details presented towards the end goal of designing controllers for a group of N differential drive robots. Multiattribute utility functions are constructed in the decision analysis literature as an integral step in accurately representing the utility for decisions involving multiple attributes and uncertainty. For more information, see [35] and the references reported therein. A multiattribute utility function exists. Many of the methodologies have presented in Section 2.1. Multiattribute utility functions represent the complement values of the remaining attributes. In order to construct a multiattribute utility function from individual utility functions, a copula will be used. Details and a formal definition of a copula will be presented in Section 2.1. Multiattribute utility functions with the following properties will be considered in the scope of this paper: continuous, bounded, nondecreasing with respect to each attribute, and for each attribute \( v_i \), there is at least one value of the complement values such that the function is strictly increasing in \( v_i \). In the formulations considered for this paper, attributes of the multiattribute utility copulas will represent the objectives of the robots. The assumptions made regarding the multiattribute utility functions are reasonable and will become clear in the subsequent definitions of the various objectives.

2.1 Multiattribute Utility Functions

Several methodologies for the creation of multiattribute utility functions exist. Many of the methodologies have forms which incorporate single-attribute utility assessments yet also make strong assumptions regarding the independence of the attributes. This limits the form of the multiattribute utility functions to those of a multilinear combination

\[
U(w_1, \ldots, w_n) = \sum_{i=1}^{n} a_i U_i(w_i) + \sum_{i=1}^{n} \sum_{j=i+1}^{n} a_{ij} U_i(w_i) U_j(w_j) + \cdots + a_{12 \ldots n} U_1(w_1) U_2(w_2) \cdots U_n(w_n),
\]

or under stricter assumptions, the multiplicative form

\[
U(w_1, \ldots, w_n) = \frac{1}{\beta} \left( \prod_{i=1}^{n} [1 + \beta a_i U_i(w_i)] - 1 \right).
\]

In these equations, the \( a_i, a_{ij}, \ldots, a_{12 \ldots n} \) and \( \beta \) parameters are weights chosen by the multiattribute utility function designer. Throughout this discussion, attributes are represented by the various objective functions. More information regarding their incorporation into controllers designed using a multiattribute utility copula will be presented in Section 3.4. Accordingly, methodologies that relax the independence assumptions have been developed to allow for the construction of multiattribute utility copulas with more general forms. Perhaps the methodology allowing for creation of the most general class of multiattribute utility functions is that of multiattribute utility copulas. Multiattribute utility functions constructed through the use of copula structures allow for the creation of a broad class of utility functions as highlighted in [35]. This class of utility functions must satisfy a set of properties originally required for the construction of multivariate probability functions from univariate marginal distributions. Some of these properties can be relaxed through the use of utility copulas allowing for the creation of an expanded class of multiattribute utility functions.

Sometimes referred to as Sklar copulas, [36] notes that copulas were initially used to construct multivariate probability functions and in the development of a version of the triangle inequality suitable for probability theory. These copula structures can be used to create joint probability functions by combining univariate marginal probability functions. For more information on copula structures, see [35, 36] and the references therein. An \( n \)-dimensional copula \( C \) is defined as follows:

**Definition 1.** An \( n \)-dimensional copula is a function \( C \) with the following properties:

- The domain of \( C \) is \( I \times I \times \cdots \times I = I^n \) where \( I = [0, 1] \) and \( 0 \leq C(w_1, \ldots, w_n) \leq 1 \).
- For every \( w_i \in I \):
  \[ C(w_1, \ldots, w_{i-1}, 0, w_{i+1}, \ldots, w_n) = 0, \quad (3) \]
  and \( C(1, \ldots, 1, w_i, 1, \ldots, 1) = w_i, \quad i = 1, \ldots, n. \quad (4) \)
- For every \( w_{i1}, w_{i2} \in I, \ i = 1, \ldots, n, \) such that \( w_{i1} \leq w_{i2} \) for \( i = 1, \ldots, n \) and where \( [w_1, w_2] = [w_{i1}, w_{i2}] \times \cdots \times [w_{n1}, w_{n2}] \)
  \[ V_I([w_1, w_2]) \geq 0. \quad (5) \]

In the properties above, \( V_I([w_1, w_2]) \) is the \( I \)-volume of the set of vertices \([w_1, w_2] \). An \( I \)-volume is defined as:
As assumed earlier, the min \( C \) properties lead to the following theorem from [36]:

Definition 2. Let \( W_1, \ldots, W_n \) be nonempty subsets of \( I \), and let \( H \) be a function with domain \( W_1 \times \cdots \times W_n \). Let \( S = [w_{11}, w_{12}] \times \cdots \times [w_{n1}, w_{n2}] \) be a rectangle whose vertices are all in the domain of \( H \) and such that \( w_{1i} \leq w_{12}, \forall i = 1, \ldots, n \). Then, the \( H \)-volume of \( S \) is given by

\[
V_H(S) = \sum \text{sgn}(c) H(c),
\]

where \( c = (c_1, \ldots, c_n) \) is a vertex of \( S \), each \( c_j \) is equal to either \( w_{1i} \) or \( w_{12} \), and

\[
\text{sgn}(c) = \begin{cases} 
1, & \text{if } c_j = w_{1i} \text{ for an even number of } j \text{'s,} \\
-1, & \text{if } c_j = w_{12} \text{ for an odd number of } j \text{'s.}
\end{cases}
\]

As assumed earlier, the \( n \)-dimensional copula \( C \) is a continuous mapping from \( I^n \) to \( I \). \( C \) is labeled grounded if it satisfies (3) and \( n \)-increasing if it satisfies (5). These properties lead to the following theorem from [36]:

Theorem 1. Let \( C \) be an \( n \)-dimensional copula. Then for every \( w_i, i = 1, \ldots, n \) in the domain of \( C \),

\[
C(w_1, \ldots, w_n) \leq \min(w_1, \ldots, w_n).
\]

Proof: Let \( (w_1, \ldots, w_n) \) be an arbitrary point in the domain of \( C \). From the definition of a copula, \( C(w_1, \ldots, w_n) \leq C(1, \ldots, 1, w_i, 1, \ldots, 1) = w_i \), for all \( i, i = 1, \ldots, n \). In light of the fact that \( C(w_1, \ldots, w_n) \leq w_i \) for all \( i, i = 1, \ldots, n \), it follows directly that \( C(w_1, \ldots, w_n) \leq \min(w_1, \ldots, w_n) \).

\[ \square \]

2.2 Multiattribute Utility Copulas

Multiattribute utility copulas were developed by [35] in the field of decision analysis to create a broad class of multivariate utility functions through the combination of marginal utility functions. These multiattribute utility copulas extend the available forms of multivariate functions created through the use of Sklar copulas.

Definition 3. A multiattribute utility copula \( C_i(w_1, \ldots, w_n) \) of \( n \) variables, \( n \in \mathbb{N} \), is defined to have the following properties:

- \( C_i \) is a continuous mapping from the \( n \)-dimensional hypercube \( [0, 1]^n \) to the interval \([0, 1]\) and is normalized such that

\[
C_i(0, \ldots, 0) = 0 \quad \text{and} \quad C_i(1, \ldots, 1) = 1.
\]

- \( C_i \) is nondecreasing with respect to each of its arguments \( v_j \).

- \( C_i \) is nondecreasing with respect to each of its arguments \( v_j \).

2.3 Archimedean Copulas

One of the relevant and important functional forms used to create multiattribute utility copulas is the extended Archimedean functional form. This functional form allows for a large variety of copulas to be constructed in a straightforward manner and these copulas possess many commonly desired independence properties. In the interest of constructing a Class 1 multiattribute utility copula, we consider an Archimedean functional form, presented in [35], constructed using a multiplicative generator:

\[
C_i(w_1, \ldots, w_n) = a \psi^{-1} \left[ \prod_{i=1}^{n} \psi(l_i + (1 - l_i)w_i) \right] + b,
\]

where

\[
a = \frac{1}{1 - \psi^{-1} \left( \prod_{i=1}^{n} \psi(l_i) \right)}, \quad b = 1 - a,
\]

and in which \( 0 \leq l_i < 1 \). The generator \( \psi(\cdot) \) is equivalently referred to as the copula’s generating function. Generating functions are assumed to be continuous and strictly increasing for all \( w \in [0, 1] \), \( \psi(0) = 0 \) and \( \psi(1) = 1 \) ensuring that \( \psi(w) \) is monotonic and that its inverse is well-defined. Although the Archimedean functional form
allows for a variety of generating functions, the primary
generating function considered was:
\[
\psi(w) = \frac{1 - e^{-\delta w}}{1 - e^{-\delta}}, \quad \delta \neq 0, \quad (15)
\]
where \(\psi^{-1}(w) = \frac{1}{\delta} \ln \left(1 - \frac{1}{w} \right)\). By varying the
parameter \(\delta\), different trade-off functions among the
attributes can be achieved. Determining all parameters of
the multiattribute utility function is achieved by selecting
the \(\delta\) parameter in accordance with preferences as well as
using indifference probability assessments.

2.4 Determining Parameters of the
Multiattribute Utility Copula

In order to compute values for the \(a\), \(b\), and \(l_i\) parameters
of the Archimedean utility copula, the following \(n\) equa-
tions from [35], one for each attribute, are used
\[
a(1 - l_i) = 1 - U(w_i^0, \bar{w}_i^0), \quad i = 1, \ldots, n. \quad (16)
\]
Used together, the equations of (14) and (16) and the re-
striction \(\sum U(w_i^0, \bar{w}_i^0) \leq 1\) uniquely define all parame-
ters of the Archimedean utility copula. Values for each
\(U(w_i^0, \bar{w}_i^0), i = 1, \ldots, n\), represent the probabilities ob-
tained using indifference probability assessments. In an
indifference probability assessment, the decision maker
considers the scenario in which they are presented with
a decision between two deals. The decision maker must
determine at which probability he or she is just indiffer-
ent between taking 1) a deal consisting of the attribute
set and corresponding utility they have already achieved
\(C(w_i^0, \bar{w}_i^0); \) or 2) a deal with a binary outcome that achieves
\(C(w_i^0, \bar{w}_i^0)\) with probability \(U(w_i^0, \bar{w}_i^0)\) or \(C(w_i^0, \bar{w}_i^0)\) with
probability \(1 - U(w_i^0, \bar{w}_i^0)\). This process determines the
tradeoffs between the attributes of a multiattribute utility
copula when it is constructed using the Archimedean
functional form. In the context of multiobjective control,
where the attributes are in fact objective functions, the
indifference probability assessments represent an intuitive
and systematic method for determining the tradeoffs be-
tween objectives.

3 Control Objectives

The ultimate goal is to design control laws that accom-
plish multiple objectives using multiattribute utility cop-
ulas. In simulation and in experiments, the robots’ kine-
matics were modeled as unicycle systems:
\[
\begin{align*}
\dot{x}_i &= u_{i1} \cos x_{i1}, \\
\dot{y}_i &= u_{i1} \sin x_{i1}, \\
\dot{\theta}_i &= u_{i2},
\end{align*}
\]
where \(i = 1, \ldots, N\), and where \(N\) is the number of robots.
Relative to a global frame, \(x_{i1}\) and \(x_{i2}\) represent the Carte-
sian coordinates of robot \(i\), \(x_{i3} \in [0, 2\pi]\) its orientation an-
gle, and \(u_{i1}\) and \(u_{i2}\) its linear and angular velocity inputs,
respectively. The unicycle model of (17) is known to accu-
rate model differential drive robots, see for example the
mobile robots in [8, 37]. Within the following objectives,
position information is assumed to be communicated be-
tween the robots. In the case of collision avoidance, po-

tion information is only required from other robots and
obstacles when they are within the computing robot’s de-
tection radius. For dynamic coverage and global coverage
objectives, it is assumed that all robots have a map of the
coverage domain \(a\) \(p\) \(r\) and can update their respective
mats after receiving another robot’s map. Additionally,
the robots are assumed to know their position relative to
the coverage domain. In regards to the proximity objec-
tive, it is assumed that past a robot’s communication ra-
dius, robot’s do not have position information of the other
robots nor do they communicate their map with the other
robots.

3.1 Collision Avoidance

Often desired in conjunction with the common objectives
of trajectory tracking, waypoint following, or dynamic cov-
erage, collision avoidance is of paramount importance
to the safety of autonomous robotic systems. A collision
avoidance task might consist of using a static obstacle to
prevent robots from entering unsuitable terrain during
search and rescue, mapping, or agricultural tasks, to
prevent damage from a newly discovered mine during a
minesweeping task, or prevent inter-robot collisions in co-
operative tasking. In the scope of problems considered
herein, collision avoidance consists of both avoiding inter-
robot collisions and collisions with static obstacles. The
objective of collision avoidance between robots \(i\) and \(j\) is
cooperaive and encoded using a value function originally
used in [14]
\[
\psi_{ij}^a(x) := \left( \min \left\{ 0, \frac{||x_i - x_j||_2^2 - R_{ij}^2}{||x_i - x_j||_2^2} \right\} \right)^2. \quad (18)
\]
The parameters \(R_{ij}\) and \(r_{ij}\) denote the detection and avoid-
ance radii, respectively, of robot \(i\) with respect to robot \(j\),
where $R_{ij} > r_{ij} > 0$. The radii are design parameters selected by considering a robot’s shape and size, sensor capabilities, and/or locomotive power. As in \((17)\), $x_i$ represents robot $i$’s position and $x_j$ robot $j$’s. $P_{ij}$ is a positive definite matrix which can be used to elongate robot $i$’s circular avoidance and detection regions relative to robot $j$. In the case of avoiding collisions with a static obstacle $k$, the collision avoidance value function for robot $i$ is

$$v_{ik}^{ca}(x) := \left( \min_{0} \left( \frac{\|x_i - x_k\|^2_p - R_{ik}^2}{\|x_i - x_k\|^2_p - r_{ik}^2} \right) \right)^2. \quad (19)$$

Similar to the case of inter-robot avoidance, $R_{ik} > r_{ik} > 0$ and $P_{ik}$ is positive definite. Care needs to be taken when selecting $P_{ik}$ to ensure that $\|x_i - x_k\|^2_p \leq r_{ik}^2$ over-bounds obstacle $k$. The collision avoidance value function for robot $i$ is a combination of the avoidance value functions with respect to other robots and static obstacles

$$v_i^c = \sum_{j=1}^{N} \sum_{k=1}^{N_i} v_{ij}^c + \sum_{k=1}^{N} v_{ik}^c, \quad j \neq i, \quad (20)$$

where $N_i$ represents the number of static obstacles. The gradient of the avoidance function between robot $i$ and robot or static obstacle $j$ is given by

$$\frac{\partial v_{ij}^c}{\partial x_i} = \begin{cases} 0, & \text{if } \|x_i - x_j\|_p \geq \bar{R} \\ 4 \frac{(\bar{R} - \|x_i - x_j\|_p)^2}{(\|x_i - x_j\|_p)^3} \bar{x}_{ij}, & \text{if } \bar{R} > \|x_i - x_j\|_p > \bar{r} \\ \text{not defined,} & \text{if } \|x_i - x_j\|_p = \bar{r} \\ 0, & \text{if } \|x_i - x_j\|_p < \bar{r} \end{cases} \quad (21)$$

with the substitutions $\bar{R} = R_{ij}$, $\bar{r} = r_{ij}$, $\bar{P} = P_{ij}$, and $\bar{x}_{ij} = (x_i - x_j)^T \bar{P}$. It is worth noting that the avoidance value function of (20) has a closed form gradient and is only locally active. Correspondingly, the gradient of the overall avoidance function is

$$\frac{\partial v_i^c}{\partial x_i} = \sum_{j=1}^{N} \frac{\partial v_{ij}^c}{\partial x_i} + \sum_{k=1}^{N_i} \frac{\partial v_{ik}^c}{\partial x_i}, \quad j \neq i. \quad (22)$$

### 3.2 Proximity and Robust Communications

Another objective relevant to multi-robot systems is that of the robots keeping proximity with each other. This may be for the purposes of encouraging flocking or swarming behavior in the performance of some other objective or for promoting robust communications. We will refer to any of these class of objectives as proximity control, though we will limit the scope of problems considered herein to those in control is desired to ensure robust communications. This task is particularly relevant when distributed multi-robot systems stand to benefit from sharing information such as mapping or surveillance data to encourage efficiency and yet are limited by communication ranges as in the case of wireless Bluetooth or IEEE 802.11 technologies. The wireless communication systems used by the robots in the experiments herein, utilizing off-the-shelf IEEE 802.11 transmitter/receivers, degraded in reliability as distance increased, especially past some cutoff distance. Though this formulation does not consider communication system performance change with respect to obstructions, including the following proximity control value function represents at least a low-fidelity, simple and intuitive method for enforcing constraints associated with inter-vehicle communications:

$$v_i^p(x_i, x_j) = \max \left\{ 0, \frac{\|x_i - x_j\|^2 - \hat{R}_{ij}^2}{R_{ij}}, \quad i \neq j \right\}. \quad (23)$$

Again $x_i$ and $x_j$ represent the positions of robots $i$ and $j$. The robots’ communication capability is modeled to degrade in a radial manner. Past the distance $\hat{R}_{ij}$, communication between robots $i$ and $j$ is lost. When communication is lost, information pertaining to the coverage control objective is not exchanged between the robots. The proximity value function for robot $i$ is a sum over the $N$ inter-robot proximity value functions

$$v_i^p = \sum_{j=1}^{N} v_i^p(x_i, x_j), \quad j \neq i. \quad (24)$$

Notably, this proximity function penalizes distances past $\hat{R}_{ij}$ between robots $i$ and $j$ but the formulation does not strictly enforce a prescribed maximum distance between the robots. This is in contrast to (20) which enforces a minimum distance between robots. The gradient of the proximity objective function (23) is

$$\frac{\partial v_i^p}{\partial x_i} = 4 \max \left\{ 0, \frac{\|x_i - x_j\|^2 - \hat{R}_{ij}^2}{R_{ij}} \right\} (x_i - x_j)^T, \quad (25)$$

with the gradient of robot $i$’s proximity objective function the sum of gradients

$$\frac{\partial v_i^p}{\partial x_i} = \sum_{j=1}^{N} \frac{\partial v_i^p}{\partial x_i}, \quad j \neq i. \quad (26)$$

### 3.3 Dynamic Coverage

The primary objective of the autonomous robots in this paper is that of dynamically covering a given compact domain. Specifically, covering the domain entails sensing or
Fig. 1. Sensing capability profile.

effecting to a specified level of satisfactory coverage all points within the domain. Sensing is accomplished by any of a variety of sensors, possibly denoted by electro-optical cameras or a broad class of sensors that measure different types of radiation. Effectors may include water or chemical dispersion mechanisms used to extinguish fires, mower blades and other agricultural tools, or effectors used to mitigate the dispersion of unwanted chemicals such as oil booms and skimmers. Sensing/effecting capabilities of the robots were modeled as follows:

\[ S_i(p) = \frac{M_i}{\tilde{R}_i} \max \left\{ 0, \tilde{R}_i^2 - p \right\}^2, \quad p = \|x_i(t) - \tilde{p}\|^2. \tag{27} \]

Here, \( M_i \) represents the peak sensing capacity and \( \tilde{R}_i \) the sensing radius of robot \( i \). The circular, normally distributed sensor model described in (27) is depicted in Figure 1. For a system of \( N \) robots, the cumulative sensing function is then given by:

\[
Q(t, \tilde{p}) = \int_0^t \left( \sum_{i=1}^N S_i(\|x_i(\tau) - \tilde{p}\|^2) \right) d\tau, \tag{28}
\]

in which \( \tilde{p} = [\tilde{x}, \tilde{y}] \in \mathbb{R}^2 \) represents a point within the coverage domain and \( x_i(\cdot) \) robot \( i \)'s position in time. Tasks that may be described by the following coverage control formulation include search and rescue, fire suppression, mapping or surveillance, environmental measurement and analysis, agricultural service, minesweeping, and oil or other chemical spill cleanup.

The coverage control objective is modeled using an area integral of the time-dependent coverage error to represent the level at which the domain of interest is satisfactorily covered:

\[
e(t) = \int_D h(C^* - Q(t, \tilde{p})))\phi(\tilde{p})d\tilde{x}d\tilde{y}, \tag{29}
\]

where \( h(z) \triangleq (\max \{0, z\})^3, D \) is the compact domain to be covered, \( C^* \) is a positive scalar representing satisfactory coverage, and \( \phi(\tilde{p}) \) is a nonnegative scalar function that can be used to include position dependent preferences or previous coverage information over the coverage domain. Within (29), the term (28) consists of a system of multiple robots whose sensing regions may overlap. In light of this fact and because the area integral of (29) is calculated over a time-dependant boundary and includes jumps at the critical intersection points of the robots’ sensing regions, [18] notes that an explicit form for the time derivative of the integral can be difficult to compute. This is worth considering for analyses that include gradient based controllers or Lyapunov stability analysis. When the robots’ sensing regions do not overlap, the following modified coverage error suffices:

\[
\dot{e}(t) = \int_D \frac{d}{dt}(h(C^* - Q(t, \tilde{p})))\phi(\tilde{p})d\tilde{x}d\tilde{y} \tag{30}
\]

\[
= - \int_D h(C^* - Q(t, \tilde{p}))) \left( \sum_{i=1}^N S_i(p) \right) \phi(\tilde{p})d\tilde{p}.
\]

Through simulation and experimentation, it has been noted that even when robot sensing regions overlap, the effect of the neglected integral terms is insubstantial.

The goal of coverage control is to drive the error term of (29) to zero, yet control effort does not appear in the modified derivative term of (30). Since this is a case of singular control, we consider a modified second derivative of (29), again neglecting terms related to the boundary and jumps:

\[
\ddot{e}(t) = \int_D h(C^* - Q(t, \tilde{p}))) \left( \sum_{i=1}^N S_i(p) \right)^2 \phi(\tilde{p})d\tilde{p} \tag{31}
\]

\[ - 2 \sum_{i=1}^N \int_D h(C^* - Q(t, \tilde{p})))S_i(p)\phi(\tilde{p})(x_i(t) - \tilde{p})^T \dot{x}_i d\tilde{p}. \]

Before conducting simulations and experiments, it was assumed that the robots would not leave the coverage domain while working to bring the coverage error to zero though they are not explicitly constrained to do so. In reality, the robots can drive the coverage error to zero by moving their sensors outside of the coverage domain. To prevent this scenario from driving the error to zero, a modified version of the coverage error derivative in (30) was used:

\[
\dot{e}(t) = - \int_D h(C^* - Q(t, \tilde{p}))) \left( \sum_{i=1}^N S_i(p) \right) \phi(\tilde{p})d\tilde{p}, \tag{32}
\]
where $S' > \sum_{i=1}^{N} M_i$. The process used to define the modified second derivative $\hat{e}(t)$ follows the process used for (31):

$$\hat{e}(t) = \int \int_D \left( h(C^* - Q(t, \tilde{p})) \left( \sum_{i=1}^{N} S_i(p) \right) \left( \sum_{i=1}^{N} S' - S_i(p) \right) \right) \phi(\tilde{p}) d\tilde{p} + 2 \sum_{i=1}^{N} h(C^* - Q(t, \tilde{p}))S_i(p)(x_i(t) - \tilde{x})^T \tilde{x}_i \phi(\tilde{p}) d\tilde{p}.$$  

(33)

A key difference is that the sign of the second term in (31) and (33) differ. With $\tilde{x}_i = [\tilde{x}_{i1}, \tilde{x}_{i2}]^T$, if the terms of (33) are labeled as $\hat{e}(t) = a_0(t) + \sum_{i=1}^{N} a_{i1}(t) + a_{i2}(t)\tilde{x}_{i2}(t)$, then we have:

$$a_0(t) = \int \int_D h(C^* - Q(\cdot)) \left( \sum_{i=1}^{N} S_i(p) \right) \left( \sum_{i=1}^{N} S' - S_i(p) \right) \phi(\tilde{p}) d\tilde{p},$$  

(34)

$$a_{i1}(t) = 2 \sum_{i=1}^{N} \int \int_D h(C^* - Q(t, \tilde{p}))S_i(p)(x_i(t) - \tilde{x})^T \tilde{x}_1 d\tilde{p},$$  

(35)

$$a_{i2}(t) = 2 \sum_{i=1}^{N} \int \int_D h(C^* - Q(t, \tilde{p}))S_i(p)(x_i(t) - \tilde{y})^T \tilde{x}_2 d\tilde{p}. $$  

(36)

Remark 1. Because the robots are given the objective of avoiding static obstacles, and therefore intend to stay outside of the avoidance regions of these obstacles, it is assumed that the avoidance regions of all static obstacles do not contain points within the compact coverage domain.

The dynamic coverage formulation based on the error term given in (29) and driven by a gradient-based controller is susceptible to situations in which the error term becomes zero and yet not all points within the coverage domain are satisfactorily covered. In order to guarantee that all points will be satisfactorily covered, an additional objective is proposed. The proposed global coverage objective is designed so that if robot $i$’s nearest uncovered point $\tilde{p}^*_i$ is within its sensing region, the global coverage objective’s influence in the multiobjective controller is inactive. Because of the mutually exclusive nature of these two objectives, the global coverage objective (37) could be appended to the dynamic coverage formulation to form one coverage objective. Separating the objectives in this work allowed for targeted testing of alternative global coverage objective formulations as well as incorporation of additional objectives in the controller. The global coverage objective is motivated by the idea that if the change in cumulative sensing for the $i$th robot, $S_i(p)$, is zero and (29) is nonzero, then the robot should be directed towards the nearest uncovered point. A point is deemed uncovered if its level of coverage is less than $C^*$. The global coverage objective and dynamic coverage objective of (29) are active at mutually exclusive times, similar to a switched control law, without a need of this objective to address the other objectives. If the dynamic coverage objective of one of the robots is in a local minimum: (1) all points within the robot’s coverage radius are satisfactorily covered, (2) not all points in the coverage domain are satisfactorily covered, and (3) the error of the dynamic coverage objective is zero. By adding the global objective, if (1)-(3) are true, the nearest uncovered point is outside of the robot’s coverage radius and the error of the global coverage objective is nonzero unless all points in the coverage domain are satisfactorily covered. Conversely, if the nearest uncovered point is within the coverage radius, it will become satisfactorily covered in finite time causing the global coverage objective error to become nonzero unless all points in the coverage domain are satisfactorily covered.

When determining a robot’s nearest uncovered point, two complications arise. First, while searching over the coverage domain at some arbitrary time, before satisfactory coverage has been achieved for all points, the set of uncovered points is dense and not necessarily convex. Determining the nearest uncovered point for robot $i$ can become part of an intractable search. This is bypassed by discretizing the coverage domain, as done in both simulation and experiments, to limit the search for the nearest uncovered point to one over a set of finite size. Another complication that persists even after discretization of the search set is that the nearest uncovered point may not be unique. This is mitigated by using the diamond-shaped search algorithm given in Algorithm 1. The algorithm begins its search within $D$ from the position of robot $i$ and selects the first uncovered point found while searching progressively further from the robot in a diamond like pattern. Notably, if the nearest uncovered point is not within a distance of $d_{max}$ from robot $i$, then the centroid of the coverage domain is taken as the nearest uncovered point for robot $i$, to encourage the robot to return to the coverage domain.

The global coverage objective for robot $i$ is defined in a decentralized manner as follows:

$$v_i^c(x_i, p^*) = \frac{1}{(d_{max})^2} \max \left\{ 0, p^* - \tilde{R}_i^2 \right\}, \quad p^* = \|x_i(t) - \tilde{x}_i\|^2. $$  

(37)

This can be thought of as the sensing inability of robot $i$’s nearest uncovered point. Mentioned previously, $d_{max}$ is a
Data: Robot $i$’s position at time $t$: $x_i(t)$, Coverage Domain $D$ and its centroid $\bar{p}_c$, Over approximation of largest distance between two points in $D$: $d_{max}$

Result: The nearest uncovered point to robot $i$: $\tilde{x}_i$

$x = x_1(t), y = x_2(t)$; $\tilde{p} = \bar{p}_c$;

for $distance = 1 \rightarrow d_{max}$ do

for $index = 0 \rightarrow distance + 1$ do

$\tilde{x} = x - distance + index, \tilde{y} = y - index$;

if $Q(t, \tilde{p}) < C'$ and $\tilde{p} \in D$ then

return $\tilde{p}$;

else

$\tilde{x} = x + distance - index, \tilde{y} = y + index$;

if $Q(t, \tilde{p}) < C'$ and $\tilde{p} \in D$ then

return $\tilde{p}$;

end

end

end

for $index = 1 \rightarrow distance$ do

$\tilde{x} = x - index, \tilde{y} = y + distance - index$;

if $Q(t, \tilde{p}) < C'$ and $\tilde{p} \in D$ then

return $\tilde{p}$;

end

else

$\tilde{x} = x + distance - index, \tilde{y} = y - index$;

if $Q(t, \tilde{p}) < C'$ and $\tilde{p} \in D$ then

return $\tilde{p}$;

end

end

end

return $\tilde{p}$;

Algorithm 1: Nearest uncovered point search algorithm.

3.4 Controller Design Using Multiattribute Utility Copulas

In previous work by [38], controllers designed to accomplish multiple objectives were based on overapproximations to the maximum function. Through Lyapunov-like analysis, controllers designed using approximations to the maximum function permit determination of sufficient conditions for accomplishing the objectives. Using the upper bound given in (8), a similar analysis strategy can be employed when designing a multiobjective controller using a multiattribute utility copula. The intuitive and systematic method of indifference probability assessments are used to determine the tradeoffs between attributes of multiattribute utility copulas and will effectively determine the controller gains. Treating objectives as attributes, the goal of this section will be to show the explicit relationship between an over-approximation of the maximum function and an n-dimensional multiattribute utility copula.

Multiattribute utility copulas require the constituent attributes to be normalized between 0 and 1. In order to use the value functions given in Equations (20), (24), (29), and (37), the following transformation is used:

$$w_{ij} = e^{-\epsilon_{ij}}, v_{ij} \in [0, \infty), w_{ij} \in [0, 1]$$

∀$i, j, i = 1, \ldots, N, j = 1, \ldots, n$. (39)

By substituting the normalized objective functions of (39) into a general Sklar copula, and noting the bound in (8), the following relationship can be established

$$\max (v_{i1}, \ldots, v_{in}) \leq - \ln (C(w_{i1}, \ldots, w_{in})).$$

(40)

The bound given in Theorem 1 can be extended to the case of a multiattribute utility copula by first recalling the relationship between a Sklar copula $C$ and a multiattribute utility copula $C_i$

$$C_i(w_{i1}, \ldots, w_{in}) = a_i(C(l_{i1} + (1 - l_{i1})e^{-\epsilon_{i1}}, \ldots, l_{in} + (1 - l_{in})e^{-\epsilon_{in}}) + b_i,$$

with $0 \leq l_{i1}, \ldots, l_{in} < 1, a_i = 1/(1 - C(l_{i1}, \ldots, l_{i2})), b_i = 1 - a_i$. Using direct substitution, the following upper bound between robot $i$’s $j$th value function and the multiattribute utility copula $C_i$ holds

$$v_{ij} \leq - \ln \left( \frac{C_i(w_{i1}, \ldots, w_{in}) - 1 + a_i(1 - l_{ij})}{a_i(1 - l_{ij})} \right) = \epsilon_{ij}.$$ (42)

The value for $\epsilon_{ij}$ is given, and when $v_{ij} \leq \epsilon_{ij}$, the $j$th objective of robot $i$ is satisfied.
with the design and analysis benefit of closed form solutions. Challenges associated with computing optimal control laws were bypassed by instead making use of control-Lyapunov functions and differential inequalities. For more information about using differential inequalities, see [39, 40]. By making use of the analysis culminating in equation (41), multiattribute utility copulas can be used as over-approximations of the maximum function. If \( \rho(x) \) is used to denote the goal function of robot \( i \), then the proposed goal function which is based on a multiattribute utility copula is of the form:

\[
\rho_i(x) = -\ln (C_1(w_{i1}(x), \ldots, w_{in}(x))). \tag{43}
\]

The goal function for robot \( i \) is comprised of all of its normalized objectives. With the goal function being an over-approximation of the maximum of the objective functions, the state of robot \( i \)'s objectives are related to the following goal function definition:

**Definition 4.** A goal with corresponding goal function \( \rho_i(\cdot) \) is said to be accomplished at time \( T \) if a trajectory \( x_i(\cdot) \) of system (17) satisfies \( \rho_i(T, x_i(T)) \leq \varepsilon \) for a given value of \( \varepsilon \) and for all \( t \in [0, T] \), with \( \|x_i(0) - x_j(0)\|_{F_0}^2 > r_{ij} \) for all \( j \), \( \rho_i(t, x_i(t)) < \infty \).

Goal functions based on over-approximations to the maximum function were constructed using \( p \)-norms in previous controller designs. This facilitated the selection of \( \varepsilon \), to ensure that a goal was accomplished, based on the relationship \( \varepsilon \leq \min(e_{i1}, \ldots, e_{in}) \), with \( e_{ij} \) given in (42). By using a goal function based on a multiattribute utility copula, a different choice for \( \varepsilon \) is necessary to ensure the goal is accomplished. In the case of (43), to guarantee that all objectives of robot \( i \) are accomplished, the following choice of \( \varepsilon \) suffices:

\[
\varepsilon = \arg \min_j \{ -\ln (1 - a_i(1 - l_{ij})(1 - e^{-\varepsilon_j})) \}. \tag{44}
\]

Performance of the multi-robot system is influenced by the indifference probability assessments and the relationships given in (14) and (16) as they ultimately determine the \( l_{ij} \) parameters used in the definition of (44).

### 3.4.1 Multiattribute Utility Copula Controller

In order to accomplish robot \( i \)'s goal function, a gradient-like controller is proposed. The proposed controller aims to decrease \( \rho_i(x) \) along the trajectories of (17) in time until the goal function satisfies \( \rho_i(T, x(T)) \leq \varepsilon \) at some time \( t = T \). While the goal function is continuous and differentiable, as multiattribute utility copulas are themselves continuous and differentiable, the proposed controller design makes use of a modified form of the goal function gradient. In general, the gradient of the goal function takes on the following form due to the chain rule:

\[
\frac{\partial \rho_i(x)}{\partial x_j} = \frac{\partial \rho_i}{\partial C} \sum_{k=1}^{n} \frac{\partial C}{\partial w_{ik}} \frac{\partial w_{ik}}{\partial v_{ik}} \frac{\partial v_{ik}}{\partial x_j}, \tag{45}
\]

where \( n \) is the number of objectives for robot \( i \). For use in the multiattribute utility copula controller, the following modified form of the goal function gradient was used:

\[
\frac{\partial \rho_i(x)}{\partial x_j} = \frac{\partial \rho_i}{\partial C} \sum_{k=1}^{n} \frac{\partial C}{\partial w_{ik}} \frac{\partial w_{ik}}{\partial v_{ik}} b_{ik}, \tag{46}
\]

with \( b_{ik} \) equaling (22), (26), and (38) for the collision avoidance, proximity, and global coverage objectives respectively. In the case of the dynamic coverage objective, \( b_{ik} = -a_i(t) \), given in equations (35) and (36) with \( a_i(t) = [a_{i1}(t) a_{i2}(t)]^T \). The difference in sign for the \( b_{ik} \) for the dynamic coverage objective is a result of the fact that the gradient based controller aims to decrease the collision avoidance, proximity, and global coverage objective functions while it aims to increase the dynamic coverage objective function. The closed form control laws take on the following form:

\[
\begin{align*}
\frac{\partial \rho_i(x)}{\partial x_j} = & -k_{ij} g_{ij}^T(x_i) \left( \left\| \frac{\partial \rho_i(x)}{\partial x_j} \right\| \right) g_{ij}(x_i), \tag{47} \\
u_{ij} = & -k_{ij} \text{sign}(x_{ij} - \phi_{ij}^i), \text{ } i = 1, \ldots, N, \tag{48}
\end{align*}
\]

where \( \rho_i(x) \) is as in (46), \( k_i, k_{ij}^i > 0 \) are gains chosen to reflect robot \( i \)'s maximal translational and angular velocities, and \( \phi_{ij}^i \) is defined as:

\[
\phi_{ij}^i = \frac{\pi}{2} - \arctan \left( \frac{\partial \rho_i}{\partial x_j} \right). \tag{49}
\]

The control law in (47) is most efficient when the proportional controller in (48) drives robot \( i \)'s heading angle to the desired heading, \( \phi_{ij}^i \). In situations where \( \left\| \frac{\partial \rho_i}{\partial x_j} g_{ij}(x_i) \right\| = 0 \), a piecewise continuous control effort can be constructed by letting \( u_{ij}(x) = y_i \) where \( \|y_i\| \leq 1 \). More details regarding the existence of solutions when using admissible, piecewise continuous control laws can be found in [38].

Differential inequalities can be used to establish sufficient conditions under which the control strategy above accomplishes the goal \( \rho_i(x) \). According to the conditions given in Definition 4, what remains is to bound the time derivative of the goal function:

\[
\frac{\partial \rho_i(t, x)}{\partial t} + \frac{\partial \rho_i(t, x)}{\partial x} \frac{\partial x}{\partial t} \leq \Omega(t, x, v(t, x)), \tag{50}
\]
for any \((t, x) \in [t_0, +\infty) \times \mathbb{R}^n\) where \(\Omega(\cdot, \cdot, \cdot) : [0, \infty) \times \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}\). Because the goal function is a multiattribute utility copula which permits a closed form solution for its modified gradient (46) and in this case does not explicitly depend on time, the choice of \(\Omega(\cdot, \cdot, \cdot)\) can be simplified by using \(\bar{\rho}_i(\cdot)\). Formally, sufficient conditions for accomplishing robot \(i\)’s goals is defined in the following theorem from [38] based on theory from [39, 40]:

**Theorem 2.** If the maximal solution \(\bar{z}(t)\) of the following differential equation:

\[
\dot{z}(t) = \Omega(t, x(t), z(t)), \quad z(0) = z_0, \tag{51}
\]

with initial condition \(z_0 \geq \rho_i(0, x_0)\) satisfies inequality (50), \(\bar{z}(T) \leq \varepsilon\) for some \(T, T > 0\), and is bounded for all \(t \in [0, T]\) then the goal is satisfied according to Definition 4.

**Proof.** The maximal solution satisfies the property in (51) from which it follows that \(\rho_i(t, x(t)) \leq \bar{z}(t)\) for any \(t\). Because \(\bar{z}(T) \leq \varepsilon\), the relationship \(\rho_i(T, x(T)) \leq \bar{z}(T) \leq \varepsilon\) holds and it follows that the goal is achieved at \(t = T\). \(\square\)

### 4 Simulations

Simulations were conducted to demonstrate the effectiveness of the control laws designed using the multiattribute utility copula constructed in (13). Simulations consisted of several scripts that were developed from the ground up in MATLAB, originally for MATLAB R2011a, and used the Symbolic Math Toolbox to calculate gradients of the objective functions. Robot states were propagated with a basic difference equation with the step size nominally set to 0.1s. The script files were designed to support additional robots and for each robot to select any subset of the defined objectives. The simulation did not take advantage of the distributed nature of the controllers presented herein, computing all robot states and coverage maps and doing all data plotting on a single machine. Running on an Intel Core i7 with 8 GB of memory, simulation run times approached the simulated time for three robots with two static obstacles. This could be improved by making the coverage domain discretization coarser but was an impetus to move to a distributed multi-robot tested. An example simulation is presented in Figures 2-6. A system of three robots with kinematics modeled as in (17) were given the objective of dynamically covering a square domain with a side length of 10, units normalized, as well as the global coverage objective given in (37). Importantly, the two-dimensional coverage domain was discretized in both dimensions at an interval of 0.1 units length resulting in the coverage domain being specified by discretized points corresponding to the elements of a square array of dimension 101x101. Points within the coverage domain were to be covered until reaching the satisfactory coverage level of \(\bar{C}^* = 1\). All three robots were modeled as having the same sensing capabilities, modeled by the equation (27), with sensing radii \(\bar{R}_i = 1\) and peak sensing capacities \(M_i = 1\). For the algorithm used to calculate the robots nearest uncovered points, \(d_{\text{max}} = 10\sqrt{2}\), which corresponds to the diagonal of the coverage domain, and the coverage domain centroid was \(\bar{c}_c = (5, 5)\). The robots were given initial conditions placing them near the northwest corner of the coverage domain, with robot 1 at (8,10), robot 2 at (10,8), and robot 3 at (10,10). The robots trajectories for the duration of the simulation and relative to the coverage domain and obstacles are depicted in Figure 2.

![Fig. 2. Robot trajectories and obstacle positions relative to the coverage domain.](image-url)

The robots were also given the objective of avoiding collisions with one another as well as two elliptical obstacles, denoted with indices 4 and 5, respectively, located at (3,5) and (7,2). Avoidance and detection regions for inter-robot collision avoidance were assumed to be circular, while for collision avoidance between robots and static obstacles, the obstacles were shaped as ellipses with scaling matrices \(P_{ik} = P_{ij} = \text{diag}(1,0.5)\). Avoidance and detection radii were specified as though the robots were homogeneous with \(r_{ij} = r_{ik} = 0.1\) and \(R_{ij} = R_{ik} = 0.6\). Distances for inter-robot and robot and obstacle pairs for the duration of the simulation are given in Figure 3 and Figure 4, respectively. Notably, the robots do not come within a distance less than or equal to the avoidance radius with respect to
Christopher G. Valicka, Richard A. Rekoske, Dušan M. Stipanović, and Ali E. Abbas

The robots were all given the objective of remaining proximal to all other robots so as to model constraints for robust communications. The communications capabilities of the robots were assumed homogeneous and given a proximity radius of $R_{ij} = 8$. Within this distance, robots were allowed to share their respective coverage maps. Communication capabilities were further assumed to allow for sharing of robot coverage maps as long as the network graph of communicating robots is connected.¹ Each robot is modeled as a vertex in the graph where direct communication between a robot pair is represented by an edge and a path consists of one or more edges. It was also assumed that communication between the robots was instantaneous. The robots were limited to maximum translational and rotational velocities of $k_i = k^0_i = 1$. To determine the parameters of the multivariate utility copula, the $a_i$, $b_i$, and $l_{ij}$ parameters, indifference probability assessment values of $U_{i1} = 0.3$, $U_{i2} = 0.65$, $U_{i3} = 0.01$, and $U_{i4} = 0.01$ for collision avoidance, proximity, dynamic coverage, and global coverage, respectively were used for each robot. A value of $\delta = 1$ was used for the generating functions (15) used to create the multivariate utility copulas. An example of the coverage error is given in Figure 5 to show how the coverage error decreased in time according to the coverage map of robot 1. At a time of approximately 214s the robots had satisfactorily covered the entire coverage domain and finished in the locations shown in Figure 6. The robots’ goal functions are plotted in Figure 7. Notably, the goal functions are not strictly decreasing with respect to time. The robots’ goal functions are denoted according to the same colors as in Figure 2. Tables 1 and 2 summarize the parameters used in the simulation where units are dimensionless unless otherwise specified.

4.1 Exploration of Copula Parameter Effects on Performance

To further explore the effects the multiattribute utility copula parameters have on simulation performance, a series of thirty simulation trials were conducted. The number of simulated robots was held constant at three, with the first eight parameters of Table 1 used, and also held constant, in the subsequent trials. These parameters describe the coverage domain as well as the sensing, communications, and avoidance ranges of the robots. The indifference probability assessments, $U_{i1}$, $U_{i2}$, $U_{i3}$, and $U_{i4}$ as well as $\delta$ were varied and their values for the respective trials are given in Table 3. All simulations were run for a maximum of 500s, simulated time. In general, the parameter $\delta$ was varied through four different values over a set of indifference probability assessments, which were common across the robots. These sets of indifference probability assessments were varied to consider alternative priorities amongst the objectives. Amongst these variations, sets of four simula-

¹ In graph theory, a graph is connected if every pair of vertices is connected by a path. A path consists of at least one edge.
Table 1. Table of example simulation parameters.

<table>
<thead>
<tr>
<th>Robot</th>
<th>( C^r )</th>
<th>( \hat{R}_i )</th>
<th>( M_i )</th>
<th>( d_{max} )</th>
<th>( \hat{p}_i )</th>
<th>( x_{io} )</th>
<th>( k_i )</th>
<th>( k_i^\theta )</th>
<th>( U_{i1} )</th>
<th>( U_{i2} )</th>
<th>( U_{i3} )</th>
<th>( U_{i4} )</th>
<th>( \delta )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>-</td>
<td>-</td>
<td>10 ( \sqrt{2} )</td>
<td>(5,5)</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>(8,10, ( \frac{1}{2} ) ( \pi ))</td>
<td>1</td>
<td>1</td>
<td>0.3</td>
<td>0.65</td>
<td>0.01</td>
<td>0.01</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>(10,6, ( \frac{1}{2} ) ( \pi ))</td>
<td>1</td>
<td>1</td>
<td>0.3</td>
<td>0.65</td>
<td>0.01</td>
<td>0.01</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>(3,5, ( \frac{1}{2} ) ( \pi ))</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>(7,2, ( \frac{1}{6} ) ( \pi ))</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

Table 2. Table of pairwise simulation parameters.

<table>
<thead>
<tr>
<th>Robot</th>
<th>Robot</th>
<th>( r_{ij} )</th>
<th>( R_{ij} )</th>
<th>( \hat{R}_{ij} )</th>
<th>( P_{ij} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>0.1</td>
<td>0.6</td>
<td>8</td>
<td>diag(1.1)</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>0.1</td>
<td>0.6</td>
<td>8</td>
<td>diag(1.1)</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>0.1</td>
<td>0.6</td>
<td>-</td>
<td>diag(1.0.5)</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>0.1</td>
<td>0.6</td>
<td>-</td>
<td>diag(1.0.5)</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>0.1</td>
<td>0.6</td>
<td>8</td>
<td>diag(1.1)</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>0.1</td>
<td>0.6</td>
<td>-</td>
<td>diag(1.0.5)</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>0.1</td>
<td>0.6</td>
<td>-</td>
<td>diag(1.0.5)</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>0.1</td>
<td>0.6</td>
<td>-</td>
<td>diag(1.0.5)</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>0.1</td>
<td>0.6</td>
<td>-</td>
<td>diag(1.0.5)</td>
</tr>
</tbody>
</table>

The effects the multiattribute utility copula parameters had on several simulation performance metrics are summarized in Table 4. The metrics presented in Table 4 include the duration of the simulation in simulated time, the percentage of the coverage domain that was satisfactorily covered, the percentage time all robots remained within their proximity radius of each other, and whether there were any collisions. Considering the Archimedean function form given in (13) and the generating function chosen in (15), as \( \delta \) moved closer to zero, in general, the multiattribute utility copula utility was higher given a set of objective function values than when \( \delta \) was larger. A lower \( \delta \) value seemed to correspond to a looser coupling of objective values in situations when they competed, as the proximity and global objective objectives often did. The fastest time to cover the domain occurred in trial 3 where the proximity objective was the objective with the highest indifference to probability. The lowest coverage percentage after 500s occurred in trial 18 where all objectives were preferred the same and with higher indifference probability, as in scenario 3 mentioned above. In general, the robots seemed to satisfactorily cover the entire domain when they did not remain proximal, or even spent less time proximal, through the entire simulation. Also in general, when the dynamic coverage or global coverage objectives had high indifference probability assessment values, the multi-robot system seemed to fully cover the domain slower and less often. The indifference probability assessment for the avoidance objective seemed to have the least significant effect on the simulation performance. From this simulation data, when designing multiattribute utility copula controllers using the equations in (47) and (48), a reasonable strategy may be to consider or quantify how certain objectives will conflict with each other. Then, ensure they have different indifference probability assessment values if conflicts are common, lower the value of \( \delta \). Given the large number of parameter combinations, possible numbers of robots, and their potential effects on system evolution, only a very small subset of scenarios have been explored. Future simulations should be conducted to consider the effects of robot numbers on the system performance, each objective should be studied through dedicated scenarios, and variations amongst the robots’ indifference probability assessments should be studied.

Fig. 5. Coverage error in time according to robot 1.
Table 3. Results from simulations exploring effect of copula parameter variation.

<table>
<thead>
<tr>
<th>Trial</th>
<th>$U_{11}$</th>
<th>$U_{12}$</th>
<th>$U_{13}$</th>
<th>$U_{14}$</th>
<th>$\delta$</th>
<th>Trial</th>
<th>$U_{11}$</th>
<th>$U_{12}$</th>
<th>$U_{13}$</th>
<th>$U_{14}$</th>
<th>$\delta$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.3</td>
<td>0.65</td>
<td>0.01</td>
<td>0.01</td>
<td>2</td>
<td>16</td>
<td>0.24</td>
<td>0.24</td>
<td>0.24</td>
<td>0.24</td>
<td>0.5</td>
</tr>
<tr>
<td>2</td>
<td>0.3</td>
<td>0.65</td>
<td>0.01</td>
<td>0.01</td>
<td>3</td>
<td>17</td>
<td>0.24</td>
<td>0.24</td>
<td>0.24</td>
<td>0.24</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>0.01</td>
<td>0.9</td>
<td>0.01</td>
<td>0.01</td>
<td>1</td>
<td>18</td>
<td>0.24</td>
<td>0.24</td>
<td>0.24</td>
<td>0.24</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>0.01</td>
<td>0.9</td>
<td>0.01</td>
<td>0.01</td>
<td>0.5</td>
<td>19</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>0.01</td>
<td>0.9</td>
<td>0.01</td>
<td>0.01</td>
<td>2</td>
<td>20</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
<td>0.5</td>
</tr>
<tr>
<td>6</td>
<td>0.01</td>
<td>0.9</td>
<td>0.01</td>
<td>0.01</td>
<td>3</td>
<td>21</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>0.01</td>
<td>0.01</td>
<td>0.9</td>
<td>0.01</td>
<td>1</td>
<td>22</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
<td>3</td>
</tr>
<tr>
<td>8</td>
<td>0.01</td>
<td>0.01</td>
<td>0.9</td>
<td>0.01</td>
<td>0.5</td>
<td>23</td>
<td>0.01</td>
<td>0.01</td>
<td>0.45</td>
<td>0.45</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>0.01</td>
<td>0.01</td>
<td>0.9</td>
<td>0.01</td>
<td>2</td>
<td>24</td>
<td>0.01</td>
<td>0.01</td>
<td>0.45</td>
<td>0.45</td>
<td>0.5</td>
</tr>
<tr>
<td>10</td>
<td>0.01</td>
<td>0.01</td>
<td>0.9</td>
<td>0.01</td>
<td>3</td>
<td>25</td>
<td>0.01</td>
<td>0.01</td>
<td>0.45</td>
<td>0.45</td>
<td>2</td>
</tr>
<tr>
<td>11</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
<td>0.9</td>
<td>1</td>
<td>26</td>
<td>0.01</td>
<td>0.01</td>
<td>0.45</td>
<td>0.45</td>
<td>3</td>
</tr>
<tr>
<td>12</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
<td>0.9</td>
<td>0.5</td>
<td>27</td>
<td>0.45</td>
<td>0.01</td>
<td>0.01</td>
<td>0.45</td>
<td>1</td>
</tr>
<tr>
<td>13</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
<td>0.9</td>
<td>2</td>
<td>28</td>
<td>0.45</td>
<td>0.01</td>
<td>0.45</td>
<td>0.45</td>
<td>0.5</td>
</tr>
<tr>
<td>14</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
<td>0.9</td>
<td>3</td>
<td>29</td>
<td>0.45</td>
<td>0.01</td>
<td>0.01</td>
<td>0.45</td>
<td>2</td>
</tr>
<tr>
<td>15</td>
<td>0.24</td>
<td>0.24</td>
<td>0.24</td>
<td>0.24</td>
<td>1</td>
<td>30</td>
<td>0.45</td>
<td>0.01</td>
<td>0.01</td>
<td>0.45</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 4. Table of simulation performance metrics.

<table>
<thead>
<tr>
<th>Trial</th>
<th>Time</th>
<th>% Cov</th>
<th>% Prox</th>
<th>Col</th>
<th>Trial</th>
<th>Time</th>
<th>% Cov</th>
<th>% Prox</th>
<th>Col</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>500s</td>
<td>99.9</td>
<td>75</td>
<td>no</td>
<td>16</td>
<td>333s</td>
<td>100</td>
<td>58</td>
<td>no</td>
</tr>
<tr>
<td>2</td>
<td>297s</td>
<td>100</td>
<td>91</td>
<td>no</td>
<td>17</td>
<td>296s</td>
<td>100</td>
<td>52</td>
<td>no</td>
</tr>
<tr>
<td>3</td>
<td>147s</td>
<td>100</td>
<td>83</td>
<td>no</td>
<td>18</td>
<td>500s</td>
<td>64.8</td>
<td>71</td>
<td>no</td>
</tr>
<tr>
<td>4</td>
<td>170s</td>
<td>100</td>
<td>82</td>
<td>no</td>
<td>19</td>
<td>500s</td>
<td>88.1</td>
<td>93</td>
<td>no</td>
</tr>
<tr>
<td>5</td>
<td>160s</td>
<td>100</td>
<td>81</td>
<td>no</td>
<td>20</td>
<td>344s</td>
<td>100</td>
<td>68</td>
<td>no</td>
</tr>
<tr>
<td>6</td>
<td>184s</td>
<td>100</td>
<td>81</td>
<td>no</td>
<td>21</td>
<td>493s</td>
<td>100</td>
<td>83</td>
<td>no</td>
</tr>
<tr>
<td>7</td>
<td>500s</td>
<td>99.9</td>
<td>83</td>
<td>no</td>
<td>22</td>
<td>276s</td>
<td>100</td>
<td>69</td>
<td>no</td>
</tr>
<tr>
<td>8</td>
<td>344s</td>
<td>100</td>
<td>66</td>
<td>no</td>
<td>23</td>
<td>222s</td>
<td>100</td>
<td>100</td>
<td>no</td>
</tr>
<tr>
<td>9</td>
<td>500s</td>
<td>87.0</td>
<td>92</td>
<td>no</td>
<td>24</td>
<td>500s</td>
<td>79.5</td>
<td>100</td>
<td>no</td>
</tr>
<tr>
<td>10</td>
<td>500s</td>
<td>95.3</td>
<td>71</td>
<td>no</td>
<td>25</td>
<td>500s</td>
<td>82.3</td>
<td>100</td>
<td>no</td>
</tr>
<tr>
<td>11</td>
<td>500s</td>
<td>82.4</td>
<td>100</td>
<td>no</td>
<td>26</td>
<td>500s</td>
<td>74.2</td>
<td>100</td>
<td>no</td>
</tr>
<tr>
<td>12</td>
<td>500s</td>
<td>85.2</td>
<td>100</td>
<td>no</td>
<td>27</td>
<td>219s</td>
<td>100</td>
<td>100</td>
<td>no</td>
</tr>
<tr>
<td>13</td>
<td>500s</td>
<td>97.2</td>
<td>100</td>
<td>no</td>
<td>28</td>
<td>500s</td>
<td>79.5</td>
<td>100</td>
<td>no</td>
</tr>
<tr>
<td>14</td>
<td>500s</td>
<td>85.3</td>
<td>100</td>
<td>no</td>
<td>29</td>
<td>500s</td>
<td>82.5</td>
<td>100</td>
<td>no</td>
</tr>
<tr>
<td>15</td>
<td>286s</td>
<td>100</td>
<td>66</td>
<td>no</td>
<td>30</td>
<td>500s</td>
<td>74.6</td>
<td>100</td>
<td>no</td>
</tr>
</tbody>
</table>

Fig. 6. Level of coverage in the coverage domain with final robot locations.

Fig. 7. Robot goal functions versus time.
5 Experimental Results

Experiments were conducted to further assess the effectiveness of a multiattribute utility copula controller on a multi-robot system. The corresponding experiments demonstrate the ability of the controller to be easily implemented on a multi-robot system and the capability of the controller to ensure multiple objectives are accomplished simultaneously.

5.1 Multi-robot Testbed

Software implementing control laws based on goal functions constructed using multiattribute utility copulas was developed for and run on four-wheeled car-like robots equipped with various sensors and custom electronics. By making use of a commercial off the shelf indoor motion capture system, the multi-robot system had access to position and orientation information of all cooperating robots within the testbed. Additional software was developed to document experimental data and provide a real-time visualization of robot distances, coverage domain information, and other data relevant to the multi-robot system objectives.

Experiments were conducted in the Mechatronics Laboratory of the department of Industrial and Enterprise Systems Engineering at the University of Illinois at Urbana-Champaign. Within the lab space, the OptiTrack motion capture system from Natural Point was utilized to determine and broadcast position and orientation data for the multi-robot system. The testbed is pictured in Figure 8, with the square coverage domain outlined in masking tape containing four robots and two silver cake pans which were used as obstacles. Position and orientation information from the motion capture system was broadcast at a rate of 100 Hz and unique reflective marker arrangements on top of the robots were used by the motion capture system to provide individualized robot attitude information. An arrangement of eighteen infrared cameras formed the motion capture volume in which any robot with predefined marker arrangements could be localized. The motion capture software and a snapshot of the camera placement used to create the motion capture volume are pictured in Figure 9. This information was broadcast over a local network to all robots as well as a computer displaying a real-time visualization of robot positions in the coverage domain and its corresponding level of coverage. Figure 10 diagrams the flow of shared information for the multi-robot system.
le less modems, an OMAP-L138 dual core processor with the ARM core running embedded Ångström Linux and the DSP core running DSP BIOS, an additional TMS320F28335 processor used for motor and other low-level control, and encoders to measure robot position and orientation. The localization data produced by the encoders and the motion capture system was fused within a Kalman filter to extend the range of localization information and increase position and orientation accuracy. The four-wheeled, differential drive robots are driven using two DC motors powered by a pair of lithium polymer batteries. An example of the robots, their associated electronics, and motion capture marker arrangement used in the multi-robot experiments is presented in Figure 11. Robots did all control and objective function calculations using their onboard processors. This included inter-robot and inter-obstacle distance calculations, dynamic coverage map updating and fusion, and the search algorithm used in the global coverage objective. The robots were equipped with approximately 2 GB of flash storage to which robot position and coverage maps were saved for later analysis.

Coverage and distance information was then broadcast to the visualization software depicted in Figure 12. This software was run on a separate computer for the purpose of displaying the simulated sensing of the multi-robot system to the experiment operator. For the experiments considered herein, the robots were not equipped with a specific sensor or effector but instead, the sensing capability was simulated using the sensor model given in (27). The robots were displayed in the visualization software using unique colors. Also displayed therein are the avoidance, detection, and proximity radii of all robots, in black, as well as indicators of each robot’s respective nearest uncovered point, colored the same as the corresponding robot. Coverage information was displayed using a heat map to represent the level of coverage within the coverage domain where red was used to denote satisfactory coverage and blue to represent no coverage, in the same manner as the plots produced in the simulations. Also included for reference was the time duration of the coverage mission in the bottom righthand corner. Data collection consisted of the dedicated visualization software, data logging scripts run onboard the robots processors, and videos taken of the motion capture volume and visualization software.

5.2 Experiments

A series of experiments were conducted with various numbers of robots, obstacles, initial conditions, and controller gains. Two specific examples are presented herein with a three robot experiment denoted as Experiment 1 and a four robot experiment denoted as Experiment 2. Videos of both experiments which illustrate the robots’ behaviors are included in the corresponding Extensions, 1 and 2. For
the four robot experiment, robots were initially positioned around the northeast corner while the three robot experiment started with robots placed just north of the northern most obstacle. The initial placement of the robots for both experiments was determined by ensuring that the robots were not within each other’s or the obstacles’ avoidance regions. Furthermore, the robots were placed to be within communication range of all other robots. In general, collision avoidance followed the definition (20). Obstacles were circular to model the physical obstacles used in the testbed and avoidance radii for the static obstacles differed from the avoidance radii for the robots. Robot trajectories relative to the coverage domain and static obstacles for Experiment 1 are pictured in Figure 13. The same information for Experiment 2 is presented in Figure 14.

For both Experiment 1 and Experiment 2, the detection radii were $R_{ij} = 0.65 \text{ m}$ for inter-robot avoidance and $R_{ik} = 0.5 \text{ m}$ for robot-obstacle avoidance. The avoidance radii were $r_{ij} = 0.25 \text{ m}$ for both inter-robot and robot-obstacle avoidance. In both experiments, the proximity objective was described as in (23) with the robots desiring to stay proximal to all other robots. Distances between the robot pairs, also in relation to the detection, avoidance, and proximity radii, for Experiment 1 and 2 are pictured in Figures 16 and 15, respectively. Similarly, robot-obstacle distances for the two experiments are given in Figures 18 and 17. The distance plots demonstrate that during the entirety of both experiments, the robots did not enter the avoidance regions of each other or the static obstacles. These plots also show that the proximity objective formulation does not strictly enforce a maximum distance. It is interesting to note that in general, the robots stayed proximal, especially in Experiment 1, and in both experiments the robots tended to move away from each other very shortly after another robot or static obstacle entered a robot’s detection radius. Again, the multiattribute utility copula controllers followed the definitions given in (43) for the goal function, with the robots again modeled as unicycle systems. Accordingly, lateral velocity input was defined as in (47), with the goal function gradient modification of (46), and angular velocity input as in (48) and $\phi_0^i$ as in (49). The corresponding gains for the robots were $k_i = 0.7$ and $k_\theta^i = 1$. The copula was based on the Archimedean functional form given in (13) with a generating function (15) parameter chosen arbitrarily to be $\delta = 1$.

The testbed’s coverage domain, outlined in masking tape in Figure 8, was defined by a square with an approxi-
mate length of 4.5 meters. In software, the domain was discretized into a grid of approximately 160x160 points with each point having a dimension of 0.00079 m$^2$. A proximity radius of 3 m was selected for both experiments presented herein and identical indifference probability assessments were used for both experiments. For the collision avoidance, proximity, dynamic coverage, and global coverage objectives, the corresponding utility values were $U_{11} = 0.01$, $U_{12} = 0.97$, $U_{13} = 0.001$, and $U_{14} = 0.01$. These parameters were selected to reflect the preference of dynamic coverage over everything else, with collision avoidance and global coverage also highly preferred over proximity. For the dynamic coverage objective, each robot used the search algorithm defined in Algorithm 1 with $d_{max} = 6.364$ m defined as the diagonal of the coverage domain. In both experiments, the circular obstacles, denoted in black, were placed in roughly the same locations as noted in the trajectory plots of Experiment 1, Figure 13, and Experiment 2, Figure 14. In Experiment 1, the robots satisfactorily covered the entire coverage domain after approximately 149 s. For the Experiment 2, the robots had satisfactorily covered the entire coverage domain at a time of 97 s. The parameters of Experiment 1 are summarized in Tables 5 and 6, Experiment 2 in Tables 7 and 8.

### 5.3 Comments on the Multiattribute Utility Copula Controller and Global Coverage Objective

Through various simulations and experiments, the strengths and weaknesses of using gradient-like control laws based on goal functions constructed using multiattribute utility copulas became apparent. Copula based control laws mitigated the challenges associated with accurately determining the tradeoffs between objectives. For example, by using the indifference probability assessment values given in Section 2.4, the influence of the proximity objective on the multi-robot system behavior was reduced in a straightforward manner. By requiring the probability value to be large, the other objectives were allowed to take precedence in situations where the objectives competed with each other. By following the design procedures for multiattribute utility copulas, the search for an appropriate tradeoff value was simplified by looking in the set [0 1]. The multiattribute utility copula stands as a promising alternative to other multiattribute utility function formulations. The primary advantage over other multiattribute utility formulations is that multiattribute utility copulas decouple the multiobjective scalarization weightings from the preferences associated with the individual objectives. This manifested itself further, especially...
in the experiments, by allowing for specification of objective function parameters based solely on robot or sensor capabilities and not on weightings relative to other objectives. Furthermore, the intuitive and systematic procedure for determining the tradeoffs amongst the objectives allowed for a timely incorporation of the additional global coverage objective \((37)\) as well as additional robots. Other scalarizations require recomputing all weights when a new objective is added, increasing the effort associated with tuning the corresponding controller. For multiattribute utility copulas in practice, consideration should be made when using objective functions, especially in discrete implementations, which may regularly reach large values. Numerical limits associated with using the transformations in \((39)\) can lead to situations in which an objective effectively becomes inactive in the multiattribute utility copula controller as \(w_{ij} = 0\). This can be mitigated by considering the range of the corresponding value function \(v_{ij}\), possibly pre-scaling its values, and by considering the time step of the discrete control law implementation.

The global coverage objective allowed for distributed control of a multi-robot system without the need for a leader robot with superior dynamic coverage capabilities. It also allowed for situations in which it is impractical, by cost or sensing capability, to have a leader robot capable of sensing the entire coverage domain. Encoding the objective in this manner \((37)\) also represents an alternative to the switching control law used in [24]. The search for the nearest uncovered point, given in Algorithm 1, represents a basic but potentially inefficient method for ensuring the entire coverage domain is satisfactorily covered. It differs from the methodology for coordinated global coverage in [3] which coordinates a global coverage control using a more sophisticated and centralized scheme. What is lost in efficiency may, however, be regained in the ease of implementation. While the global coverage objective does not interfere with the dynamic coverage objective, depending on the proximity objective formulation used, it may be quite likely that the global coverage and proximity objectives conflict with each other. A potential way to overcome performance problems associated with this conflict is to modify the formulation used for the proximity objective. Specifically, instead of defining a robot’s objective as the sum over all robots, relax this formulation to be an \(\arg \min\) over the robot indices, effectively requiring that a robot be proximal to only a single, different robot.

### 6 Conclusion

Constraints and behaviors for multi-robot systems related to collision avoidance, robust communications, and dynamic coverage were encapsulated through the use of nonnegative definite objective functions. A novel objective function used to promote global coverage in a distributed fashion was presented. The multiattribute utility copula, a tool used in the field of multiattribute decision analysis, was adapted for use in multi-objective control because of its ability to intuitively and systematically assign preferences amongst attributes. Preliminary
Multiattribute utility copula functional forms. Additional objective function formulations, and alternate exploring different sensing models and implementations, multi-robot experiments presented. Future work includes was demonstrated through their use in the simulation and ease with which such control laws can be implemented provided through the application of theory from differentives, constituting a goal function. Formals for verifying that all objectives, constituting a goal function, are accomplished were provided through the application of theory from differential inequalities and Lyapunov-like analysis. The relative ease with which such control laws can be implemented was demonstrated through their use in the simulation and multi-robot experiments presented. Future work includes exploring different sensing models and implementations, additional objective function formulations, and alternate multiattribute utility copula functional forms.

### Table 7. Table of Experiment 2 parameters.

<table>
<thead>
<tr>
<th>Robot</th>
<th>C</th>
<th>R_i</th>
<th>M_i</th>
<th>d_{max}</th>
<th>\bar{p}_C</th>
<th>\bar{x}_{10}</th>
<th>k_i</th>
<th>k_i^0</th>
<th>\delta</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>-</td>
<td>-</td>
<td>6.4m</td>
<td>(2.5, 2.5)</td>
<td>-</td>
<td>-</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0.65</td>
<td>1</td>
<td>-</td>
<td>-</td>
<td>(1.25, 4.5, 3\pi/4, \pi)</td>
<td>0.7</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0.65</td>
<td>1</td>
<td>-</td>
<td>-</td>
<td>(2.5, 4.5, 3\pi/4, \pi)</td>
<td>0.7</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0.65</td>
<td>1</td>
<td>-</td>
<td>-</td>
<td>(4, \pi)</td>
<td>0.7</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>(7, \pi)</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>(7, \pi)</td>
<td>-</td>
<td>-</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Table 8. Table of Experiment 2 pairwise parameters.

<table>
<thead>
<tr>
<th>Robot</th>
<th>Robot</th>
<th>r_{ij}</th>
<th>R_{ij}</th>
<th>\bar{R}_{ij}</th>
<th>P_{ij}</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>0.25m</td>
<td>0.65m</td>
<td>3m</td>
<td>diag(1,1)</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>0.25m</td>
<td>0.65m</td>
<td>3m</td>
<td>diag(1,1)</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>0.25m</td>
<td>0.65m</td>
<td>3m</td>
<td>diag(1,1)</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>0.25m</td>
<td>0.5m</td>
<td>-</td>
<td>diag(1,1)</td>
</tr>
<tr>
<td>1</td>
<td>6</td>
<td>0.25m</td>
<td>0.5m</td>
<td>-</td>
<td>diag(1,1)</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>0.25m</td>
<td>0.65m</td>
<td>3m</td>
<td>diag(1,1)</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>0.25m</td>
<td>0.65m</td>
<td>3m</td>
<td>diag(1,1)</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>0.25m</td>
<td>0.5m</td>
<td>-</td>
<td>diag(1,1)</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>0.25m</td>
<td>0.5m</td>
<td>-</td>
<td>diag(1,1)</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>0.25m</td>
<td>0.65m</td>
<td>3m</td>
<td>diag(1,1)</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>0.25m</td>
<td>0.5m</td>
<td>-</td>
<td>diag(1,1)</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>0.25m</td>
<td>0.5m</td>
<td>-</td>
<td>diag(1,1)</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>0.25m</td>
<td>0.5m</td>
<td>-</td>
<td>diag(1,1)</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>0.25m</td>
<td>0.5m</td>
<td>-</td>
<td>diag(1,1)</td>
</tr>
</tbody>
</table>

### Appendix: Index to Multimedia Extensions

<table>
<thead>
<tr>
<th>Extension</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Video</td>
<td>Experiment 1 from start to finish showing the three robots' trajectories and obstacles next to the visualization software</td>
</tr>
<tr>
<td>2</td>
<td>Video</td>
<td>Experiment 2 from start to finish showing the four robots' trajectories and obstacles next to the visualization software</td>
</tr>
</tbody>
</table>

**Acknowledgement:** This work has been supported by the National Science Foundation under grant CMMI 12-58482.

### References


