# Multi-Robot Control for Circumnavigation of Particle Distributions

Sarah Tang, Dylan Shinzaki, Christopher G. Lowe, and Christopher M. Clark

**Abstract** In this work, we present a decentralized controller for the tracking and following of mobile targets, specifically addressing considerations of: 1) not altering target behavior, 2) target states represented by multiple hypotheses, and 3) limited information from bearing-only sensors. The proposed controller drives a team of $n$ robots to circumnavigate an arbitrary distribution of target points at a desired radius from the targets. The controller also dictates robot spacing around their circular trajectory by tracking a desired relative phase angle between neighbors. Simulation results show the functionality of the controller for arbitrary-sized teams and arbitrary stationary and moving particle distributions. Additionally, the controller was implemented on OceanServer Iver2 AUVs. Tracking results demonstrate the controller's capability to track a desired radius as well as maintain phase with respect to a second AUV.

## 1 Introduction

In recent years, multi-robot systems have been developed for a variety of applications. Compared to single robot systems, they have better spatial-temporal coverage, greater manipulation force, and are more robust to mission failure. In particular, they can be applied to tracking problems, where a team of robots attempt to gather in-

Sarah Tang
Princeton University, Princeton, NJ, USA, e-mail: sytang@princeton.edu

Dylan Shinzaki
Stanford University, Stanford, CA, USA, e-mail: shinzaki@stanford.edu

Christopher G. Lowe
CSU Long Beach, Long Beach, CA, USA e-mail: clowe@csulb.edu

Christopher M. Clark
Harvey Mudd College, Claremont, CA, USA e-mail: clark@hmc.edu



The final publication is available at Springer via https://doi.org/10.1007/978-3-642-55146-8_11.

formation about or follow one or more targets. Yet while these teams can be more effective, they also come with a number of challenges. The robots themselves may have nonholonomic kinematics. Sensor information might be limited, noise and external disturbances could make available sensor information hard to process, and maintaining multiple hypotheses as well as fusing sensor information are nontrivial design challenges. In addition, the team often has to maintain a certain distance from their targets so as to not influence natural behavior.

This work is motivated by the scientific goal of autonomous tracking and following of long migratory fish species (e.g. sharks) using Autonomous Underwater Vehicles (AUVs). Previous work [6] has shown the ability of a single AUV equipped with bearing-only sensors to track and follow tagged leopard sharks using a Particle Filter (PF) for estimating target states. In moving towards tracking species whose movement behaviors would be altered by the presence of tracking unit, a multi-AUV controller is required. This controller needs to enable active state estimation using multiple sensor vantage points to reduce the effects of bearing-only sensors, while simultaneously ensuring that all AUVs maintain some predetermined distance from the target.

The remainder of this section will present related work. Section 2 will define the problem at hand in terms of four subproblems and detail our proposed solutions. Section 3 will present experimental results from a controller implementation, and Section 4 will conclude the paper.

### *1.1 Background*

Numerous approaches have been proposed to the general problem of multi-robot control. Control methods are either centralized, where a central processor controls the entire team, or decentralized, where each robot maintains its own controller using sensor information about other robots and its environment. A number of techniques have been proposed for both methods.

A behavior-based controller maintains different modes of action that, depending on the goal of the robot, are weighed differently to give rise to different group dynamics. For example, Balch and Arkin [2] appropriately weight motor schemas for moving to destination, obstacle avoidance, and waypoint navigation. Similarly, Bougherty et al. [5] implement the basic behaviors of move-to-goal, avoid-obstacle, maintain-relative-distance, maintain-relative-angle, and stop, to form the control laws for their robotic team. Generally, such behavior-based methods are successful at maintaining desired group dynamics, however, are hard to analyze for stability guarantees.

The leader-following approach designates robots in the team as leaders or followers. Leader robots simply travel the desired trajectory of the team while follower robots are responsible for maintaining group formation by tracking desired bearing angles and distances to designated leaders. For example, Desai et al. [4] describe decentralized control laws that drive a follower to either maintain distance and bear-

ing to one leader or two bearings to two leaders. Such methods tend to be simpler in implementation, but they are heavily dependent on effective communication between robots. The system is also prone to single-point failure, and the leader robot typically has no means of detecting or compensating for lost followers.

With a virtual structure approach, a group of virtual positions are arranged in a rigid desired formation and moved along a desired trajectory. Each robot follows the specific path traced out by one of the virtual positions. This method is used by Young et al. [17], who drive robots to their desired virtual positions with a Proportional-Derivative (PD) control law, and Tan and Lewis [9], who apply this technique to a vision-based tracking system. One big advantage of such methods is that it is relatively straightforward to specify desired behavior and analytically guarantee stability.

These control methods have proven effective in the domain of multi-robot tracking. Liu et al. [10] propose behavioral-based control that uses reinforcement learning to adjust behavior weights for a target-tracking controller, Chung et al. [3] implement a gradient-based decentralized control law for dynamic target tracking, and Mazo et al. [11] use the virtual vehicle approach. In addition to these main approaches, Lee et al. [8] explore target tracking control for unicycle mobile robots using a nonlinear state feedback controller, while Papanikolopoulos et al. [15] propose a method where output of a visual tracking system serve as input to a Proportional-Integral (PI) controller, pole assignment controller, or an Linear-Quadratic-Gaussian (LQG) controller. However, these methods track a only single estimate of either one or more moving targets.

A few methods do combine control with state estimation. Mottaghi and Vaughan [12] use a Particle Filter to create a potential field for a virtual structure based robotic controller, and Wang et al. [16] propose a flocking control method for multiple robots moving to an estimated position from a distributed Kalman filter. However, these methods drive robots directly to estimated destinations rather than circumnavigation. To accommodate for limited sensing, Zhou et al. [18] propose an iterative Gauss-Seidel Relaxation (GSR) algorithm to drive robots to the best sensing location for a team of heterogeneous robots tracking a moving target, accounting for limited measurements and motion constraints (e.g. maintaining a minimum distance from the target). Paley [14] propose control methods for circular motion of robots moving in external flow fields, however, only allows for circling of one center point. Lan et al. [7] propose a hybrid control method to capture targets with circular motion around the target, but does not combine this method with state estimation.

In our work, we use a stability-guarantee based approach to develop a distributed control method that will simultaneously accommodate all three of the previously mentioned issues, namely, accounting for multiple hypotheses, limited sensing, and maintaining distance from the target. To this end, we use a proportional control law to drive arbitrary-sized teams to circumnavigate target distributions at a specified radius from all targets and angle offset from each other.

## 2 Multi-Robot Circumnavigation

The aim of this work is to design decentralized control laws to drive a team of $n$ robots to circumnavigate any distribution of $p$ target points. There are two main objectives: First, each robot should circumnavigate all targets at a constant desired radius. Second, each robot should maintain a constant phase difference between itself and the robots in front and behind it. Spacing robots apart in this manner helps avoid collisions and enables information gain from different sensor vantage points with minimal overlap. Fig. 1 illustrates this ideal configuration with four robots circling clockwise around a single target point.

To accomplish this design task, the problem is broken down into four subproblems: 1) design a single robot controller that drives the robot to circumnavigate a single target, 2) design a multi-robot controller that extends the previous controller to enable robots to track a desired phase difference as they circumnavigate a single target, 3) design a target allocation system that further extends the controller to multiple targets, and 4) design an ordering algorithm that initializes each robot's relative position with respect to the circumnavigation loop. Preliminaries are first presented followed by a proposed solution to each of the four subproblems.

### *2.1 Preliminaries*

In this work, a team of $n$ robots operate within a 2D obstacle-free workspace. Each robot $i$ is represented at time $t$ by a state vector $\mathbf{X}_{i,t}^{robot}$ comprised of its position, $(x_{i,t}, y_{i,t})$, and yaw orientation, $\theta_{i,t}$, with respect to an inertial cartesian coordinate frame:

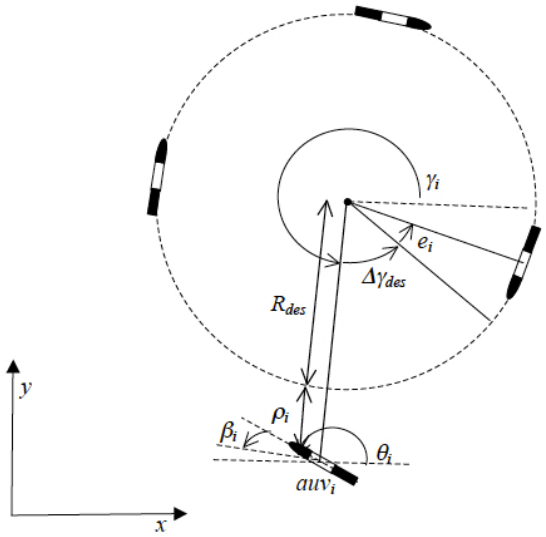



**Fig. 1** This picture exemplifies the desired circling behavior, with a team of four AUVs circling one target point. The AUVs circle the target at a desired $R_{des}$ and are spaced around the circle with a desired phase difference $\Delta\gamma_{des}$. The error variables $\beta$, $\rho$, and $e$, presented in Eq. 8 - 10, are also illustrated for AUV $i$.

$$\mathbf{X}_{i,t}^{robot} = [x_{i,t}\ y_{i,t}\ \theta_{i,t}]^T. \tag{1}$$

The kinematics of the system are assumed to follow the first order discrete time equations in Eqs. 2 - 4. In this case, $v_{i,t}$ and $\omega_{i,t}$ are the forward and rotational velocities of the $i^{th}$ robot, respectively, and serve as the robot control inputs.

$$x_{i,t+1} = x_{i,t} + v_{i,t}\cos(\theta_{i,t})\Delta t \tag{2}$$
$$y_{i,t+1} = y_{i,t} + v_{i,t}\sin(\theta_{i,t})\Delta t \tag{3}$$
$$\theta_{i,t+1} = \theta_{i,t} + \omega_{i,t}\Delta t \tag{4}$$

To facilitate control with respect to a target at position ($x_{target,t}$, $y_{target,t}$), three additional states are defined. First, let $r_{i,t}$ be the distance between the target and robot $i$ at time $t$. Second, let $\gamma_{i,t}$ be the relative bearing angle of robot $i$ relative to the target. Third, let $\theta_{des_i,t}$ be the desired yaw angle for robot $i$, which is in a direction tangent to a circle centered on the target.

$$r_{i,t} = \sqrt{(x_{i,t} - x_{target,t})^2 + (y_{i,t} - y_{target,t})^2} \tag{5}$$
$$\gamma_{i,t} = \tan^{-1}\left((y_{i,t} - y_{target,t})/(x_{i,t} - x_{target,t})\right) \tag{6}$$
$$\theta_{des_i,t} = \gamma_{i,t} - \frac{\pi}{2} \tag{7}$$

Given a desired radial distance from all targets, $R_{des}$, and a desired relative phase offset, $\Delta\gamma_{des}$, the system can now be described in terms of error variables $\rho_{i,t}$, $\beta_{i,t}$, $e_{i,t}$.

$$\rho_{i,t} = \mathrm{R}_{des} - r_{i,t} \tag{8}$$
$$\beta_{i,t} = \theta_{des_i,t} - \theta_{i,t} \tag{9}$$
$$e_{i,t} = \Delta\gamma_{des} - (\gamma_{i,t} - \gamma_{i-1,t}) \tag{10}$$

The term $e_{i,t}$ represents the error in phase between two vehicles $i$ and $i-1$. Although $e_{i,t}$ is indexed with a robot index $i$, it actually represents the angle difference between a pair of robots, and so, there is no $e_{0,t}$ term. These errors form the state vector associated with each robot:

$$\chi_{i,t} = [\rho_{i,t}\ \beta_{i,t}\ e_{i,t}]^T. \tag{11}$$

Fig. 1 illustrates these variables for a sample tracking scenario. The kinematics of these controlled variables can be derived from substituting the derivatives of Eqs. 5 - 7 into the derivatives of Eqs. 8 - 10:

$$\rho_{i,t+1} = \rho_{i,t} + v_{i,t}\sin(\beta_{i,t})\Delta t, \tag{12}$$

$$\beta_{i,t+1} = \beta_{i,t} + \left(-\frac{v_{i,t}\cos(\beta_{i,t})}{R_{des}-\rho_{i,t}} - \omega_{i,t}\right)\Delta t, \tag{13}$$

$$e_{i,t+1} = e_{i,t} + \left(\frac{v_{i,t}\cos(\beta_{i,t})}{R_{des}-\rho_{i,t}} - \frac{v_{i-1,t}\cos(\beta_{i-1,t})}{R_{des}-\rho_{i-1,t}}\right)\Delta t. \tag{14}$$

## *2.2 Single Robot Circumnavigation of a Single Target*

With the preliminaries presented above, the first design problem can stated as follows: Given a robot $i$ that behaves according to Eqs. 2 - 4, determine the discrete time control update laws for the control vector:

$$\mathbf{U_{i,t}} = [v_{i,t}\ \omega_{i,t}]^T \tag{15}$$

that drive errors $\rho_{i,t}$, $\gamma_{i,t}$, and $e_{i,t}$ to 0 as time $t$ approaches infinity.

In the single robot case, the robot is assumed to travel at some nominal velocity $v_{i,t} > 0$. The rotational velocity, $\omega_{i,t}$, is used to drive each robot towards its desired path around the target. The proposed control law is shown in Eq. 16, where $R_{des}$ is the desired radius of the circle centered on the target being tracked, $K_\beta$ and $K_\rho$ are proportional control gains, and $\Delta t$ is the time step in seconds between control signal updates.

$$\omega_{i,t} = -\frac{v_{i,t}\cos(\beta_{i,t})}{R_{des}-\rho_{i,t}} + \frac{K_\beta}{\Delta t}\beta_{i,t} + \frac{K_\rho}{\Delta t}\rho_{i,t} \tag{16}$$

Note that the first term in Eq. 16 is a feedback linearization term to accommodate the nonlinear dynamics in Eq. 13. The singularity in Eq. 16 is avoided by initializing robots such that they do not invoke the controller to begin circumnavigation until they are within some minimum $\rho_o$ of all targets. Thus, for all $t \geq 0$, $\rho_{i,t} \leq \rho_o < R_{des}$. This initialization scheme is described in detail in Sec. 2.5.

To derive the system's stability conditions, the control law in Eq. 16 is substituted into Eqs. 12 and 13 to arrive at the closed-loop error dynamic equations. These equations can further be simplified using a small angle approximation $\sin(\beta_{i,t}) \approx \beta_{i,t}$. The error dynamics and corresponding eigenvalues are described in matrix formulation in Eq. 17 and Eq. 18, respectively.

$$\begin{bmatrix}\rho_i \\ \beta_i\end{bmatrix}_{t+1} = \begin{bmatrix}1 & v_{i,t}\Delta t \\ -K_\rho & 1-K_\beta\end{bmatrix}\begin{bmatrix}\rho_i \\ \beta_i\end{bmatrix}_t \tag{17}$$

$$\lambda = \frac{(K_\beta - 2) \pm \sqrt{K_\beta^2 - 4K_\rho v_{i,t}\Delta t}}{2} \tag{18}$$

The discrete system is asymptotically stable if and only if all eigenvalues have magnitude less than 1. The problem is not fully constrained and has many solutions, but for $\Delta t > 0$ and positive gains, gain and velocity bounds are given in Eqs. 19 - 21.

$$K_\rho > 0 \tag{19}$$

$$0 < K_\beta < 4 \tag{20}$$

$$\frac{2(K_\beta - 2)}{K_\rho \Delta t} < v_{i,t} \leq \frac{K_\beta^2}{4K_\rho \Delta t} \tag{21}$$

## *2.3 Multi-Robot Circumnavigation of a Single Target*

In the multi-robot case, the controller in Eq. 16 is used to set $\omega_{i,t}$ for each robot, while $v_{i,t}$ is modulated to drive phase errors $e_{i,t}$ to 0 while still ensuring it lies within the bounds defined by Eq. 21. The following controller for $v_{i,t}$ is proposed:

$$v_{i,t} = \frac{R_{des} - \rho_{i,t}}{\mathrm{R}_{des}\cos(\beta_{i,t})}\left(v_{nom} + \frac{R_{des}K_\gamma}{\Delta t}(e_{i+1,t} - e_{i,t})\right). \tag{22}$$

Substituting Eq. 22 into Eq. 14 yields the closed loop kinematic equation for $e_{i,t+1}$. As $e_{i,t}$ represents the error between two vehicles, the $e_{i,t}$ term is dropped for the $0^{th}$ vehicle and the $e_{i+1,t}$ term is dropped for the $n-1^{th}$ vehicle. Since $e_{i,t+1}$ is dependent on $e_{i-1,t+1}$ and $e_{i,t+1}$, the matrix formulation is required:

$$\begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ \vdots \\ e_n \end{bmatrix}_{t+1} = \begin{bmatrix} 1-2K_\gamma & K_\gamma & 0 & 0 & \cdots & 0 & 0 \\ K_\gamma & 1-2K_\gamma & K_\gamma & 0 & \cdots & 0 & 0 \\ 0 & K_\gamma & 1-2K_\gamma & K_\gamma & \cdots & 0 & 0 \\ & & & \ddots & & & \\ 0 & 0 & 0 & 0 & \cdots & K_\gamma & 1-2K_\gamma \end{bmatrix} \begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ \vdots \\ e_n \end{bmatrix}_t. \tag{23}$$

Again, all eigenvalues must have magnitude less than 1 for the system to be asymptotically stable. The eigenvalues vary with $n$; the case of three robots is presented in Eqs. 24 - 26 to exemplify how stability criteria can be used to determine appropriate gain values.

$$\begin{bmatrix} e_1 \\ e_2 \end{bmatrix}_{t+1} = \begin{bmatrix} 1-2K_\gamma & K_\gamma \\ K_\gamma & 1-2K_\gamma \end{bmatrix} \begin{bmatrix} e_1 \\ e_2 \end{bmatrix}_t \tag{24}$$

$$\lambda = 1-3K_\gamma, 1-K_\gamma \tag{25}$$

$$0 < K_\gamma < \frac{2}{3} \tag{26}$$

### 2.4 Multi-Robot Circumnavigation of Multiple Targets

The control laws can be further extended to allow circumnavigation of a set $P$ of $p$ different targets, denoted:

$$P = [x_j \; y_j \; \theta_j \; v_j \; \omega_j], j \in [1:p]. \tag{27}$$

Each robot invokes the control laws from Eqs. 16 and 22 to track the closest target in $P$ at the current time step $t$. In other words, control variables $\rho_{i,t}$, $\beta_{i,t}$, and $e_{i,t}$ are calculated with respect to only one target $j$. In this way, robots can, for example, track a set of particles from a PF state estimator [6] that represent one or many objects in the workspace.

To make transitions between targets smoother, the tracked target at time step $t$ is that which minimizes the Euclidian distance between candidate targets and the predicted robot state at $t+\tau$. Here, $\tau$ is some positive integer, and the state is predicted assuming $v_{i,k} = v_{i,t}$ and $\omega_{i,k} = \omega_{i,t}$ for $k \in [1,\tau]$. A minimal $\tau$ based on a robot's minimum turning radius can be used to ensure $\rho_{i,t+1} > 0$ when tracking a new target location (i.e. the robot will not come within $R_{des}$ of the next target).

### 2.5 Multi-Robot Ordering During Circumnavigation

The control laws presented in Eq. 16 and 22 ensure that a team of robots will eventually converge to circumnavigation of a collection of targets with relative phase tracking for collision-free motion and multiple sensor vantage points. To prevent collisions from occurring before system convergence, Alg. 1 is used to assign a static order to all robots based on their initial positions with respect to targets.

In Alg. 1, Line 1, each target from $P$ introduced in Eq. 27 is grouped into one of $C$ *clusters*. The $c$th *cluster* is a subset of the $p$ targets in $P$ that contains $p_c$ targets, where $p_c > 0$ and the Euclidean distance of each target to at least one other target in the group is $< 2R_{des}$. In the case where $C \leq n$, the number of robots assigned to each *cluster*, denoted $n_c$, is proportional to the number of targets in the *cluster*. If $C > n$, only the largest *clusters* will be assigned robots.

The following procedure is carried out within each *cluster*; for simplicity, Alg. 1 assumes only one *cluster*, so $n_c = n$ and $p_c = p$. Robots identify a circle $O$ centered at the average position of all targets, defined here as $\mathbf{X}_o = (x_o, y_o)$. The radius $r_{max}$ of $O$ is the minimum radius that encapsulates all target circumnavigation loops and robots. A set of $n$ boundary points are evenly distributed on $O$, to be used as initial destinations for the $n$ robots, as seen in Lines 4 - 6. The robots are matched to *boundary points* according to the relative bearing angles $\kappa$ from center $(x_o, y_o)$, as calculated on Lines 7-9. That is, the robot with the $k^{th}$ greatest bearing angle among robots will be matched to the *boundary point* with the $k^{th}$ greatest bearing angle among *boundary points*, as in Lines 10 - 15.

**Algorithm 1** findOrder($\mathbf{X}_{i,t}^{robot}\ \forall i = 1..n$, $\mathbf{X}_{j,t}^{target}\ \forall j = 1..p$, $R_{des}$, $t$)

1: $clusters \leftarrow$ clustering($\mathbf{X}_{i,t}^{robot}\ \forall i = 1..n$, $\mathbf{X}_{j,t}^{target}\ \forall j = 1..p$)
2: $\mathbf{X}_o \leftarrow$ average over $\mathbf{X}_{j,t}^{target}\ \forall j = 1..p$
3: $r_{max} \leftarrow \max_{\forall j=1..p}(|\mathbf{X}_{j,t}^{target} - \mathbf{X}_o| + R_{des})$
4: **for** $k$=1..$n$ robots **do**
5: $\quad \mathbf{X}_{k,t}^{boundarypoint} \leftarrow [x_o + r_{max}\cos(2k\pi/n) \quad y_o + r_{max}\sin(2k\pi/n)]^T$
6: **end for**
7: **for** $i$=1..$n$ robots **do**
8: $\quad \kappa_i \leftarrow \tan^{-1}((y_{i,t} - y_o)/(x_{i,t} - x_o))$
9: **end for**
10: $order \leftarrow$ sorted list of indices of $\kappa_i$ in ascending order from $\theta = 0$
11: **for** $i$ = 1..$n$ robots **do**
12: $\quad k \leftarrow i^{th}$ element of $order$
13: $\quad \mathbf{X}_{i,t}^{desBP} \leftarrow \mathbf{X}_{k,t}^{boundarypoint}$
14: **end for**
15: **return** $order, \mathbf{X}_{i,t}^{desBP}\ \forall i = 1..n$

Upon initialization, each robot $i$ drives directly towards its individual *boundary point* $\mathbf{X}_{i,t}^{desBP}$ and pauses motion when within a predetermined distance error $\rho_o$ of $O$, where $\rho_o < R_{des}$ as noted in Sec. 2.2. Only when all robots have distance to $O < (R_{des} + \rho_o)$ does the group start circumnavigating its designated cluster of targets. Note this algorithm is run only once, after which robot order remains constant.

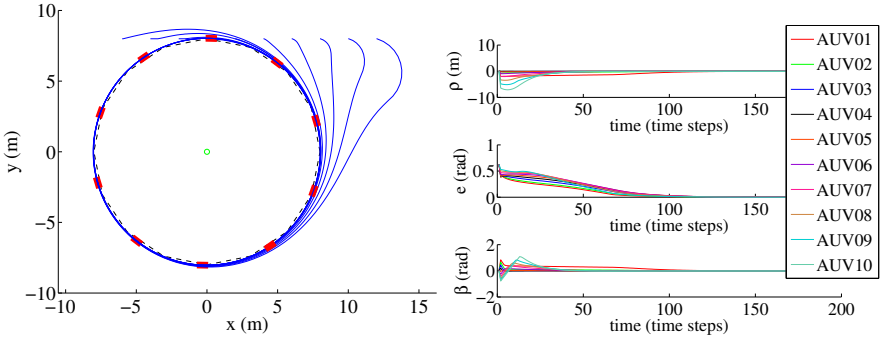
## 3 Results

The proposed control laws were simulated in Matlab to verify tracking performance. Controller gains and velocity constraints were selected to ensure controller stability as dictated by the inequalities given in Eqs. 19 - 21 and exemplified by Eqs. 24 - 26. Table 1 lists the chosen constants.

Table 1: Controller constants

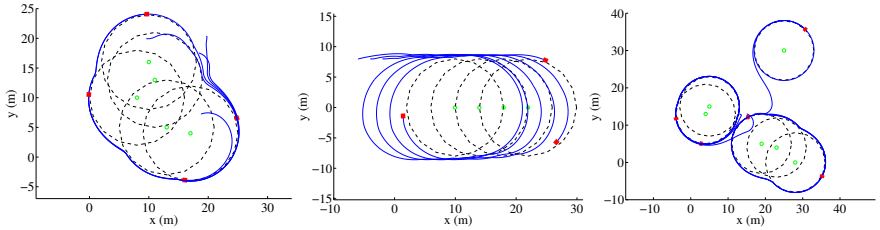
| Simulation | | | | | |
|---|---|---|---|---|---|
| $K_\rho$ | 0.3 | $K_\beta$ | 1.5 | $K_\gamma$ | 0.5 |
| $v_{nom}$ | 2.5m/s | $v_{min}$ | 1m/s | $v_{max}$ | 3m/s |
| $R_{des}$ | 8m | $\Delta\gamma_{des}$ | $\frac{\pi}{2}$ if $R = 2$, otherwise $\frac{2\pi}{n}$ | | |
| Field tests | | | | | |
| $K_\rho$ | 0.1 | $K_\beta$ | 0.4 | $K_\gamma$ | 0.5 |
| $v_{nom}$ | 0.56m/s | $v_{min}$ | 0.3m/s | $v_{max}$ | 0.8m/s |

Fig. 2a illustrates a team of 10 robots circumnavigating a single target, and Fig. 2b shows the corresponding error terms converging to zero over time. To demonstrate the versatility of the controller, Fig. 3 illustrates a variety of different scenarios involving multiple targets (e.g. particle distributions), both stationary and moving.




(a) Trajectory (b) Control variables

Fig. 2: Circumnavigation of one target by 10 AUVs




(a) 4 robots, 5 targets (b) 3 robots, 4 moving targets (c) 5 robots, 6 targets, 3 clusters

Fig. 3: Circumnavigation of various target distributions

To validate the control system on a real test platform and progress towards autonomous fish tracking with AUVs, the proposed controller was implemented on an OceanServer Iver2 AUV [13], pictured in Fig. 4a. The vehicle is approximately 1.27m in length, 0.15m in diameter, and 20kg in weight, with two fins to control pitch, two rear fins to control yaw, and a three-blade propeller to provide forward velocity. A 24V servo-controlled DC motor gives the AUV a speed range of approximately 0.5m/s to 2.0m/s. A built-in GPS receiver provides longitude and latitude measurements and a three-degree-of-freedom compass provides heading measurements. The AUV runs two processors, a main Intel 1.6 GHz ATOM processor with Windows XP and an additional low power Intel 1.6GHz ATOM processor; the first

is responsible for fundamental control of actuators and sensors, while the second is designated for external programs, such as the controller. The AUV communicates with laptop via wireless ethernet. Deployments were conducted in Carnegie Lake, Princeton, NJ and Fisherman's Cove, Santa Catalina Island, CA.

As the following experimental procedure applies to each individual AUV independently, the subscript $i$ will be dropped from future notation. To appropriately actuate the AUV, the relationships from controller outputs, $v_t$ and $\omega_t$, to actuator values for the motor and yaw fin, denoted $u_{v,t}$ and $u_{\omega,t}$, respectively, were determined. Specifically, two functions - one relating $v_t$ to $u_{v,t}$, the other relating $u_{v,t}$ and $\omega_t$ to $u_{\omega,t}$ (as the motor speed of the AUV also dictates the maximum achievable angular velocity) were sought. Position data of the AUV running at various constant $u_{\omega,t}$ and $u_{v,t}$ values were gathered. Offline, actual forward and angular velocities of the AUV were estimated as the secant line between position measurements at $t-1$ and $t-T$ for $t \in [T : t_{end}]$ and averaged to approximate the $v_t$ or $\omega_t$ corresponding to a setting of $u_{v,t}$ or $u_{\omega,t}$. The desired functions were then derived from linear regressions, where $T$ was chosen between 1 and 8 to yield the highest $r^2$ value.

Additionally, there was a clear time delay between the actuation of the AUV and actual achievement of the actuated angular velocity. In this light, the AUV's speed and angular velocity are estimated in real time, denoted $v_{est,t}$ and $\omega_{est,t}$, respectively, using a decaying weighing scheme. The real time estimate of $v_t$ provides more accurate control when used within Eq. 16, and $\omega_t$ is estimated for for offline analysis. In future work, a more sophisticated real time estimation method, such as Kalman Filtering or Extended Kalman Filtering, can be implemented.

Results from using one AUV to track various (known) target distributions are summarized in Table 2. For these experiments, $R_{des} = 8$m. Note that average and standard deviations do not increase significantly when tracking moving distributions, demonstrating the controller's stability even for tracking moving targets.

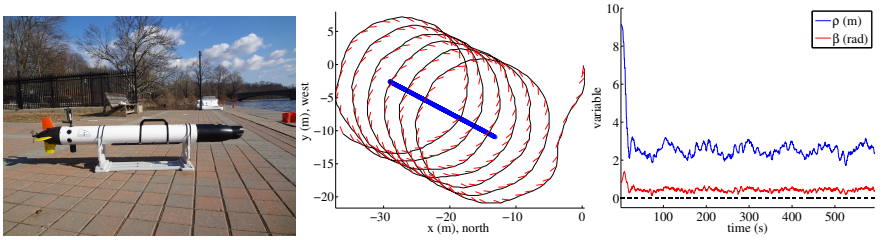
Table 2: Results for single AUV tests

| Test | $\rho_{avg}$ (m) | $\sigma_\rho$ (m) | $\beta_{avg}$ (rad) | $\sigma_\beta$ (rad) |
|---|---|---|---|---|
| single target | 2.7070 | 1.5325 | 0.4678 | 0.5598 |
| 3 targets | 2.6239 | 0.4830 | 0.4472 | 0.1386 |
| moving target | 2.6508 | 0.8504 | 0.4474 | 0.2679 |

Fig. 4 plots the AUV motion for moving target tracking; Fig. 4b plots the actual AUV trajectory in black with AUV heading every ten time steps in red while Fig. 4c plots controller variables over time. Fig. 4b shows that the AUV is able to follow and circle the target consistently, and Fig. 4c additionally shows that $\rho_t$ and $\beta_t$ decay towards steady-state values over time, through there seems to be a steady-state error present in both variables. From Table 3, average $\beta_t$ is 0.45 radians, or approximately 26 degrees, and average $\rho_t$ is 2.65 meters. An approximate three second time lag in actuation effect is a potential cause of this error, and methods to compensate for

the time delay could be explored. In addition, though $K_\beta$ and $K_\rho$ were chosen for practical performance, they might not have been optimal gains. There are several techniques for theoretically optimal gain selection [1] that could be investigated.

Note that looking at the trajectory in Fig. 4b, there seems to be a delay in GPS measurements from corresponding compass readings. The AUV's heading is not tangent to its path at its current position, but rather, to the curvature a small distance forward, suggesting that GPS coordinates are received slower than compass headings.




(a) Iver2 AUV (b) Tracking trajectory (c) Error variables

Fig. 4: Single AUV experiment

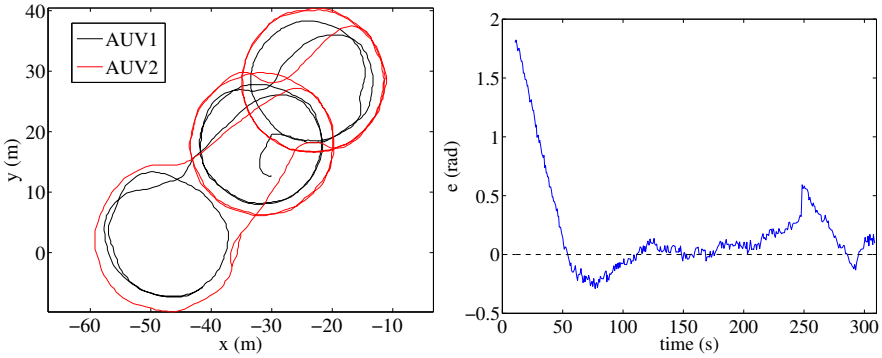
Next, the phase tracking capabilities of the controller are tested. First, a second AUV was simulated in code. At each time step, the simulated AUV was propagated forward using control law outputs and derived kinematics, and its resulting position was passed to the real AUV, assuming perfect and instantaneous communication between vehicles. For these experiments, $R_{des} = 8$m and $\Delta\gamma_{des} = \frac{\pi}{2}$. Tracking statistics are presented in Table 3. In addition to minimizing error in $\rho_t$ and $\beta_t$, the AUV is able to approach the desired $\gamma_t$ as well.

Table 3: Results for simulated multi-AUV tests

| Test | $\rho_{avg}$ (m) | $\sigma_\rho$ (m) | $\beta_{avg}$ (rad) | $\sigma_\beta$ (rad) | $\gamma_{avg}(rad)$ | $\sigma_\gamma$ (rad) |
|---|---|---|---|---|---|---|
| stationary target | 2.1773 | 0.5199 | 0.4670 | 0.1484 | -1.5796 | 0.0303 |
| moving target | 2.0083 | 0.3342 | 0.4215 | 0.1043 | -1.5505 | 0.0277 |

Additionally, experiments were done with two AUVs, with $R_{des} = 10\text{m}, 12\text{m}$ and $\Delta\gamma_{des} = \pi$. $R_{des}$ of the two AUVs were set differently to avoid collision due to malfunctions during test (e.g. an AUV running out of battery). The two AUVs tracked a single target that, rather than moving continuously, jumped to a new location after a given amount of time. This behavior simulates the possibility of sudden changes in state estimate due to new location measurements after losing measurements for a span of time. Fig. 5 illustrates the tracking trajectories. Again, it can be seen that

both AUVs successfully track their respective desired radius while driving phase error $e_t$ to 0. Note that the sudden increase in $e_t$ around 210s results from the movement of the target to a new location; the AUVs, however, are able to minimize $e_t$ again after some time.




(a) Tracking trajectory (b) $e_t$ over time

Fig. 5: Two-AUV experiment

## 4 Conclusions

In this work, we present a new multi-robot control strategy that enables multiple nonholonomic robots to circumnavigate an arbitrary distribution of targets while maintaining a constant standoff distance between robots and targets and a desired spacing between robots. A benefit of this strategy is that the control is decentralized and robots only need information regarding the positions of their neighbors. Bounds on controller gains to ensure system stability and convergence to the desired circumnavigation behavior were derived. Simulations were presented with various particle distributions and velocities. To demonstrate the applicability of the strategy to real systems, it was implemented on an AUV subject to real world disturbances (e.g. currents). In the future, this controller will be applied to a multi-AUV system and use target sensors and estimators to determine target states in real-time.

## 5 Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. 1245813.

## References


1. Åström, K.J., Murray, R.M.: Feedback Systems: An Introduction for Scientists and Engineers. Princeton University Press (2008). Available online at http://press.princeton.edu/titles/8701.html
2. Balch, T., Arkin, R.C.: Motor schema-based formation control for multiagent robot teams. In: In Proceedings of the First International Conference on Multi-Agent Systems, pp. 10–16. AAAI Press (1995)
3. Chung, T.H., Burdick, J.W., Murray, R.M.: A decentralized motion coordination strategy for dynamic target tracking. In: Proceedings of the 2006 IEEE International Conference on Robotics and Automation, ICRA 2006, May 15-19, 2006, Orlando, Florida, USA, pp. 2416–2422. IEEE (2006)
4. Desai, J., Ostrowski, J., Kumar, V.: Modeling and control of formations of nonholonomic mobile robots. IEEE Transactions on Robotics and Automation **17**(6), 905–908 (2001)
5. Dougherty, R., Ochoa, V., Randles, Z., Kitts, C.: A behavioral control approach to formation-keeping through an obstacle field. In: Aerospace Conference, 2004. Proceedings. 2004 IEEE, vol. 1, p. 175 Vol.1 (2004)
6. Forney, C. and Manii, E. and Farris, M. and Moline, M.A., and Lowe, C. G. and Clark, C.M.: Tracking of a Tagged Leopard Shark with an AUV: Sensor Calibration and State Estimation (2011)
7. Lan, Y., Yan, G., Lin, Z.: A hybrid control approach to cooperative target tracking with multiple mobile robots. In: Proceedings of the 2009 conference on American Control Conference, ACC'09, pp. 2624–2629. IEEE Press, Piscataway, NJ, USA (2009)
8. Lee, S.O., Cho, Y.J., Hwang-Bo, M., You, B.J., Oh, S.R.: A stable target-tracking control for unicycle mobile robots. In: Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems, vol. 3, pp. 1822–1827 (2000)
9. Lewis, M.A., Tan, K.H.: High precision formation control of mobile robots using virtual structures. Auton. Robots **4**, 387–403 (1997)
10. Liu, Z., Ang, M.H.J., Seah, W.K.G.: A stable target-tracking control for unicycle mobile robots. In: Proceedings of the 2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (2005)
11. Mazo, M., Jr., Speranzon, A., Johansson, K.H., Hu, X.: Multi-robot tracking of a moving object using directional sensors. In: In Proceedings of the International Conference on Robotics and Automation (ICRA, pp. 1103–1108 (2004)
12. Mottaghi, R., Vaughan, R.: An integrated particle filter and potential field method applied to cooperative multi-robot target tracking. Auton. Robots **23**(1), 19–35 (2007). DOI 10.1007/s10514-007-9028-9
13. OceanServer Technology Inc., http://www.iver-auv.com: Iver2 Autonomous Underwater Vehicle (2012)
14. Paley, D.A.: Cooperative control of an autonomous sampling network in an external flow field. In: CDC, pp. 3095–3100. IEEE (2008)
15. Papanikolopoulos, N., Khosla, P., Kanade, T.: Visual tracking of a moving target by a camera mounted on a robot: A combination of control and vision. IEEE Transactions on Robotics and Automation **9**(1), 14–35 (1993)
16. Wang, Z., Gu, D.: Cooperative target tracking control of multiple robots. Industrial Electronics, IEEE Transactions on **59**(8), 3232 –3240 (2012)
17. Young, B., Beard, R., Kelsey, J.: A control scheme for improving multi-vehicle formation maneuvers. In: American Control Conference, 2001. Proceedings of the 2001, vol. 2, pp. 704 –709 vol.2 (2001). DOI 10.1109/ACC.2001.945797
18. Zhou, K., Roumeliotis, S.: Multirobot active target tracking with combinations of relative observations. Robotics, IEEE Transactions on **27**(4), 678 –695 (2011)