BTECH PROJECT - EE496

Cooperative Circumnavigation of a Target via multiple nonholonomic UAVs

Supervisor : Dr. Anoop Jain

Arzoo Qureshi (B18EE063), Rashmi Mertia (B18EE040)

ABSTRACT

Circumnavigation is an efficient Unmanned aerial vehicle motion that gathers information about a target while orbiting around it at a predetermined distance. The objective of this paper is to first implement a control algorithm that allows this circumnavigation mission to be completed in a GPS-denied environment using only range measurement.Further,cooperative circumnavigation of a stationary as well as moving target with multiple nonholonomic robots is addressed in this paper.The multi-robot system's goal is to circle the moving target with a specified radius, circular velocity, and inter-robot angular spacing.

1 MOTIVATION

The past decade has witnessed increasing associate interest in developing uncrewed Aerial Vehicles (UAVs) for both military and civilian applications. The typical applications are intelligence, surveillance, security, and reconnaissance missions, where the objective is to track some region under inspection. Using a robot or a network of autonomous robots (UAV's) that are essentially smaller, cheaper, and require less human management than human-crewed aircraft is a more efficient way to gather information. They are deployed in such a way that it orbits around the target at the desired distance. Such UAV motion is called circumnavigation. We implement different control algorithms to achieve the circumnavigation mission.

2 Problem Description and Methodology

2.1 Part I : Static target and single UAV

We first consider a scenario where a static target is required to be circumnavigated by a robot.The main idea of our controller is to adjust the heading of the UAV such that it moves towards a well-designed circle of radius r_d around the target. To achieve this, we use range and range rate measurements. This approach guarantees global convergence regardless of the initial distance from the UAV to the target. The associated control inputs are bounded.
We characterize the relationship between the UAV states given by [x, y, ψ] and design control input ω.
Here, ψ denotes the heading of the UAV.

In the Figure 2.1 shown, T is the unknown target and we want the UAV to orbit on the black solid circle.

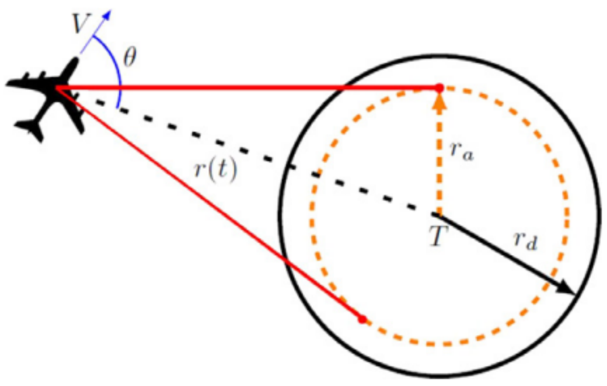


Figure 2.1. Illustrative example of variables used

The objective is to design control input ω such that $r(t) \rightarrow r_d$ as $t \rightarrow \infty$. That is to achieve a stable circular motion around the target. The speed of the UAV remains constant. The rate of r(t) is controlled by the bearing angle θ, where $\theta(t) \in [0, 2\pi)$ at time t is defined as the angle between the UAV forward direction and the direction from the UAV to the target measured counterclockwise.Thus, **r and θ are chosen as the state variables** which can then be **controlled by** ω.
The respective equations are written below.

In **cartesian coordinates** :
$\dot{x} = V \cos \psi$

$\dot{y} = V \sin\psi$
$\dot{\psi} = \omega$

Here, $[x, y]^T$: 2D location of UAV
ψ : The heading of the UAV
ω : The control input
V : constant velocity of UAV

Range measurement denoted by r =$\sqrt{(x - x_T)^2 + (y - y_T)^2}$

UAV dynamics in polar form:
$\dot{r} = -V cos\theta$
$\dot{\theta} = \omega + \frac{V\sin\theta}{r}$

The control law:

$$\omega = \begin{cases} \text{k[Vcos}(\text{arcsin}\frac{r_a}{r(t)}) - \dot{r}(t)] & \text{if } r(t) > r_a \\ 0 & \text{otherwise} \end{cases}$$

Here, k : non-zero constant gain

Now, let's see how we reached at this particular control law to achieve our desired UAV circumnavigation

For the first condition $r(t) > r_a$ the UAV is outside the circle of radius r_a(Later we will determine what exactly is this r_a) and for this case we saw that the control law is the difference of $V\cos(\text{arcsin}\frac{r_a}{r(t)})$ and $\dot{r}(t)$. Analytically this condition means that ω will try to align $\dot{r}(t)$ with $V\cos(\text{arcsin}\frac{r_a}{r(t)})$ because $V\cos(\text{arcsin}\frac{r_a}{r(t)})$ is the rate of change of r(t) when the UAV moves towards one of the two tangent points.
And if $r(t) < r_a$ then UAV has entered the undesired region so we want it to come out of that region and therefore the control law iz zero here so that UAV follows a straight line motion and come out of the circle having radius r_a

r_a is strictly smaller than r_d, which implies that the UAV should aim towards a circle with a smaller radius in order to establish the desired circular motion stable and it is given by:

$$r_a = \sqrt{r_d^2 - \frac{1}{k^2}}$$

Let's see how we arrived to this formula. Let's assume the UAV has settled finally around the target in a stable circular motion with the radius of circle as r_d. Since the velocity of UAV is V which is constant so the magnitude of angular velocity with which it will have circulae motion is $\omega = \frac{V}{r_d}$. Now from the control law we can find this ω and equate it with $\frac{V}{r_d}$ and since it will follow circular motion so r_d will be greater than r_a (because otherwise it has to follow a straight line path).

So, $|\omega|$ becomes $|-kV\cos(\arcsin\frac{r_a}{r(t)})|$ and this should be equal to $\frac{V}{r_d}$

And this is only possible if $r_d = \sqrt{r_a^2 + \frac{1}{k^2}}$

The above control algorithm can guarantee globally asymptotic convergence and the equilibrium is exponentially stable which provide an improved robustness against noise.

2.2 Part II: Static Target and Multiple UAVs

The goal for the multi-robot system is to circumnavigate a target with prescribed radius, circular velocity, and inter-robot angular spacing. In this direction, a distributed control strategy in polar coordinates is implemented for the robot team to achieve asymptotic convergence to the desired motion.

We represent the information exchange between UAVs by an undirected graph G = {V, E} where a set of vertices V={1, 2, . . . , n} indexed by the group members and set of edges E = {(i, j) V × V ‖ $j \in N_i$} representing inter-robot communications.The graph is connected.

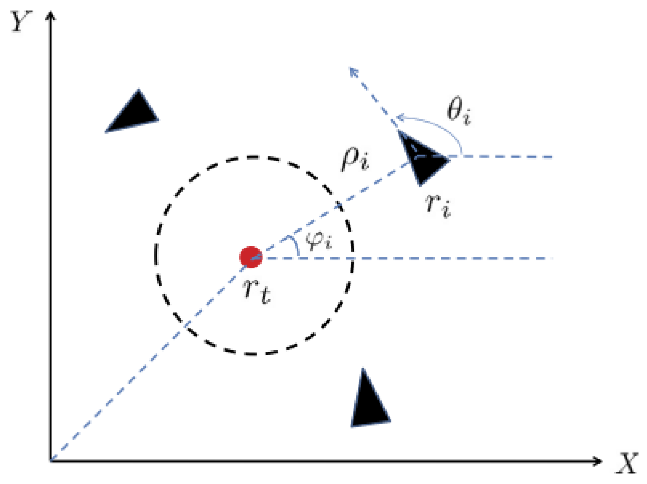


Figure 2.2

In the Figure 2.2 shown r_t is the coordinate vector of target, ρ_i is the relative distance between i^{th} robot and target, r_i is the coordinate vector of i^{th} robot, θ_i is the position and orientation of the i_{th} robot, ϕ_i is the angle between each robot and target

Given a target with position r_t and velocity $\dot{r}_t$ (if moving target), then we define the relative distance and angle between each robot and target as:

$\rho_i = ||r_i - r_t||$

$\phi_i = atan2(\frac{y_i - y_t}{x_i - x_t})$

Now the goal is to design a control law such that:

$\lim_{t\to\infty} \rho_i(t) = \rho_d \ \forall \ i \in N$

$\lim_{t\to\infty} \dot{\phi}_i(t) = \omega_d \ \forall \ i \in N$

$\lim_{t\to\infty} \phi_i(t) - \phi_j(t) = \delta_{ij} \ \forall \ i \in N, \forall \ j \in N_i$

where $\rho_d > 0$ is the prescribed radius $\omega_d \in \mathrm{R}$ is the prescribed circular velocity $\delta_{ij} = \delta_{ji} \in [-\pi, \pi]$ is the prescribed inter robot angular spacing.

Now, since we want the robots to have an angular spacing so let's take a non linear function

$h_i = h(e_i)$

where,

$e_i = \sum_{j \in N_i} a_{ij}(\phi_i - \phi_j - \delta_{ij})$

Here the function h_i is responsible for angular spacing, and it will be zero when the desired angular spacing is achieved.

The dynamics in polar coordinates:

$\dot{\rho}_i = v_i \cos(\theta_i - \phi_i) - \alpha_i$

$\dot{\phi}_i = \frac{v_i \sin(\theta_i - \phi_i) - \beta_i}{\rho_i}$

Here α_i and β_i are functions if the taget is moving

$\alpha_i = \dot{x}_t \cos\phi_i + \dot{y}_t \sin\phi_i$

$\beta_i = \dot{y}_t \cos\phi_i - \dot{x}_t \sin\phi_i$

Here, $\gamma_i = \theta_i - \phi_i$, and γ_{id} is the virtual signal of γ_i:

$$\gamma_{id} = atan2(\frac{\rho_d(\omega_d - h_i) + \beta_i}{-k_1\tilde{\rho}_i + \alpha_i})$$

The control Laws:

$$v_i = \sqrt{(-k_1\tilde{\rho}_i + \alpha_i)^2 + (\rho_d(\omega_d - h_i) + \beta_i)^2}$$

$$\omega_i = -k_2\tilde{\gamma}_i - k_3 v_i \zeta_i \tilde{\rho}_i + \dot{p}hi_i + \dot{\gamma}_{id}$$

Here ζ_i is defined as:

$$\zeta_i = \frac{\cos\gamma_i - \cos\gamma_{id}}{\gamma_i - \gamma_{id}}$$

3 Results and Discussion:

3.1 PART I:

We implemented the above mentioned control algorithm in which a UAV orbits around an unknown static target at a desired distance,in a GPS denied environment. Only range measurements were utilized to accomplish this.

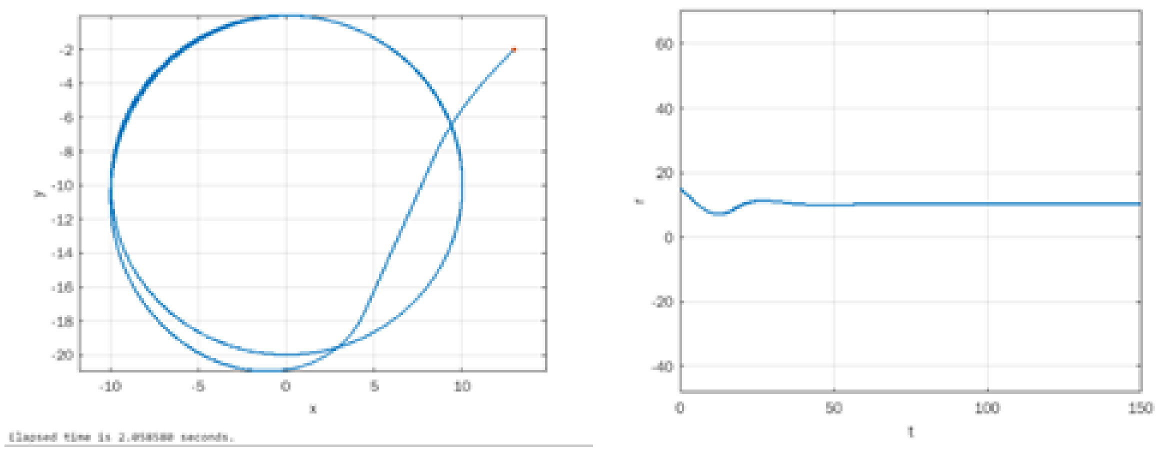
Now,for simulation purpose,the values are:
$r_d = 10, k = 0.2, V = 1$

Initial position of UAV is :
$x = 13, y = -2, \psi = \frac{5\pi}{4}$

And the initial position of target is:
$x_T = 0, y_T = -10$
r_d= 8.6603

3.1.1 SIMULATION RESULTS OF PART I:

We implemented the control algorithm by considering a point target that is safe to be around as we gather information from a distance. But it is not always the case. Sometimes, UAVs deal with targets that are dangerous and not safe to be around. Thus UAVs need to maintain a minimum safe circle around them. This minimum safe circle



```
Elapsed time is 2.058580 seconds.
```

Figure 3.1. Simulations results

is then the restricted area for the UAVs. A control algorithm can be designed. If the distance of UAVs from the target becomes less than the safe radius, the algorithm restricts the UAV from entering this safe minimum radius. If the UAV distance from the target is greater than the safe minimum circle's radius, we could follow the same control algorithm as above.

The Figure 3.2 below shows the worst case possible. Here, as the UAV enters the r_a circle, we know its ω is zero and has a velocity V=1. Before entering the minimum safe circle r_s, one of the worst possible cases is when its path is tangential to that circle. We can calculate the distance it has to travel before it comes out of the r_a circle and the time required to travel that distance. These parameters would be helpful to design other safety parameters for the UAVs.
Using geometry,we can find the distance BC or BD and it comes out to be:

BC(or BD)=$2r_a\cos(\lambda)$
where,λ is the angle between tangent BC(or BD) and r_a at point C(or D)

We assume r_s=2

Also,we can find the time needed to cover this distance with V=1

The calculated values are:
BC=16.85[units] T=16.85[units]

3.2 PART II

Here we have a multi robot system and they have to circumnavigate a the target.We consider a multi-robot system of two unicycles with an undirected chain communica-

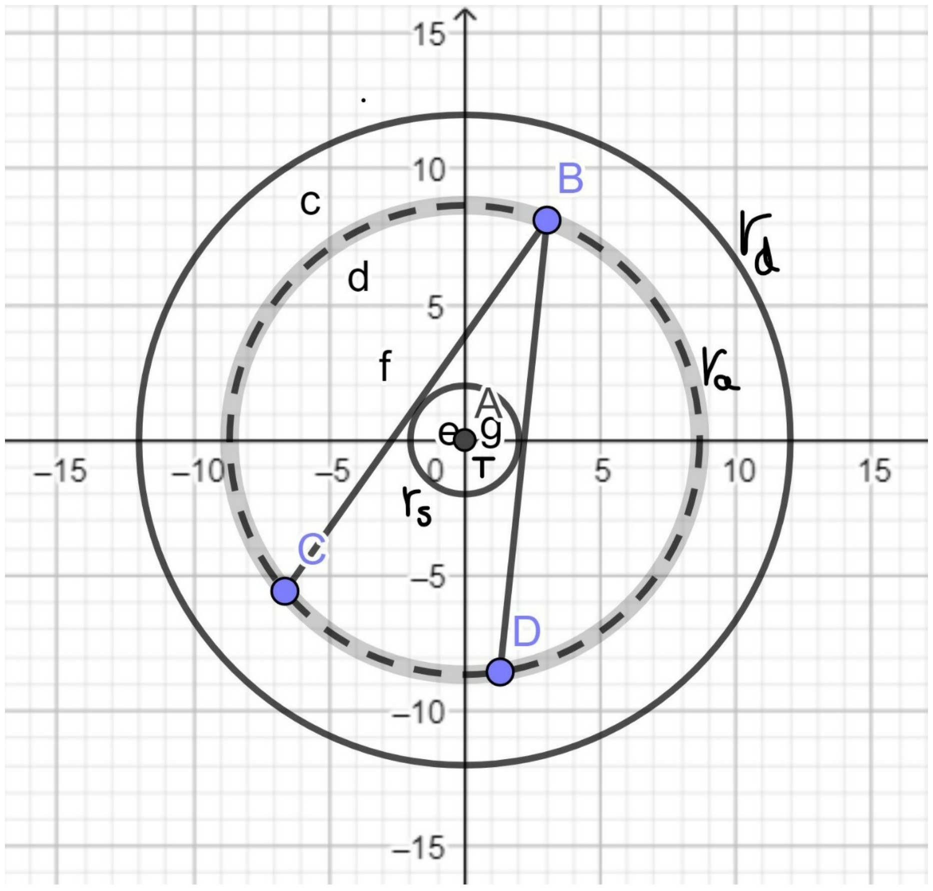


Figure 3.2. Target having minimum safe radius

tion graph.For simulation purpose,the values are:

The initial positions of the robots:
r_1=$[-6,-1]^T$
r_2=$[-7,-3]^T$

The initial position and velocity of the target:
r_t=$[0,0]^T$
$\dot{r}_t$=$[0,0]^T$

Here we have initially assumed a stationary target.We will further extend the work to a moving target.

The specified radius and circular velocity of the circle around the target are :ρ_d = 3 m, ω_d = 1 rad/s
,
The desired configuration of the robots is to maintain a circular formation with equal phasing; as a result, the inter-robot angular spacing is prescribed as δ_{12} = $2\pi/3$, δ_{21}= -δ_{12}

The nonlinear function h is chosen to be:
$h_i = 0.5\tanh(ei)$
$e_i = \sum_{j\in N_i} a_{ij}(\phi_i - \phi_j - \delta_{ij})$

The control gains are selected as:
k1 = 0.5, k2 = 2, k3 = 0.2

3.2.1 SIMULATION RESULTS OF PART II

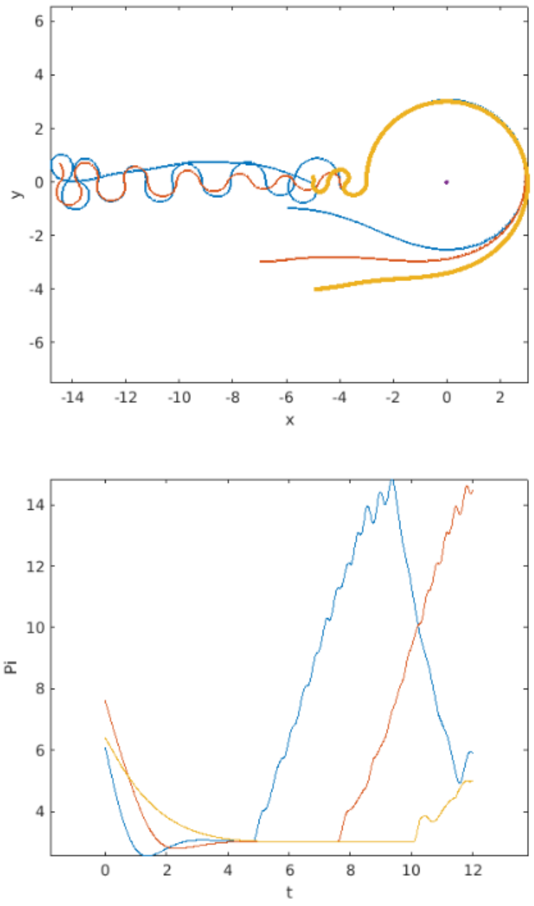


Figure 3.3. Simulation results of Part II

In part 2 results (Figure 3.3), the plots obtained are not the desired plots. We worked using Euler's formula to solve the differential equations. The Euler method is numerical for solving ordinary differential equations, provided we have the initial values. It is the most basic Runge- Kutta method. Since it is a first-order method, the local error per step here is directly proportional to the step size's square and the global error at a given time instant is directly proportional to the step size. We had the differential equations with us using geometry and control laws. We can consider the differential equation are formulas by which we can find the slope of tangent lines at any point of the curve given the coordinates of that point.

Now, since we know the initial value, we know the initial point of the curve that we want to plot. And from the differential equation, we can find the slope of the curve. We will take a tiny step from the initial point towards the tangent to reach another point, and repeating this procedure will form our curve. Since we took a tiny step, it was safe to assume that the curve slope did not change much during this small step and the next point also lies on the curve. Our initial condition will change, and the point that we have reached now will be considered as the initial value. The next slope will be calculated using this initial value.

Mathematically:

Let, $y'(t) = f(t, y(t))$ be the differential equation

and, $y(t_o) = y_o$ be the initial condition

Let the step size be h, and $t_n = t_o + nh$.Therefore, we can write:

$t_{n+1} = t_n + h$

$y_{n+1} = y_n + hf(t_n, y_n)$

Euler's method is comparatively less accurate than other higher-order Runge-Kutta methods because the error depends on lower powers of step size. Also, the Euler method can give unbalanced output because it becomes numerically unstable when the differential equations are stiff. Since we didn't get the expected results, we moved to the ode45 solver method.

The ODE45 is MATLAB's standard solver for ordinary differential equations. It implements the Runga Kutta method with a variable time step so that the computation is efficient.

The following general problem can be solved by ode45:
$\frac{dy}{dt} = f(t, y)$
$y(t_o) = y_o$

Here, the independent variable is t, and f(t,y) is a function that depends on y and t.y represent a vector consisting of dependent variables. In general, if our ode is of higher-order, then the first step should be to reduce the given ODE to a series of first-order differential equations.

Since we have a lot of equations and a lot of variables, we chose three variables x, y, and θ as our state variables. We can write the equation in terms of these state variables and their first-order forms. Finally, when we have everything in the first-order form, we can use the following command in our main code:

$[t, y] = ode45(@fname, tspan, yinit, options)$

Here, **fname** is the function name matlab file uses to find the function f(y,t).

tspan is a vector and it tells the starting and end limit of the integration and also the length of the time step that we require. For example if we want to integrate from t=0 to t=100 and we need to take 1000 time steps,therefore we write t_{span} = [0:0.1:100].

yinit represents a vector containing all the initial conditions of y vector.

The results are expected to follow the desired reults.

4 CONCLUSION

4.1 PART I

In the first part of paper, we implemented a control algorithm based on range-only measurement such that a UAV can circumnavigate an unknown target at the desired distance under a GPS-denied environment.The results came out to be the desired one.We also discussed how this control algorithm can be extended to restrict the UAV's entry in the minimum safe circle around target if the target is not safe to be in close proximity.We provided some data so that other parameters constraints can be calculated by considering the worst case possibility.This work can be extended further to find the updated control laws to solve the minimum safe circle problem and we are looking forward to solve this problem in future work.

4.2 PART II

In part II of the paper,we addressed the circumnavigation problem involving multiple nonholonomic robots and a target.Initially we used Euler's formula that has constant step size and error depends on lower powers of step size.The results we got didn't match the desired results and we moved to ODE45 method to solve the differential equations.It's variable time step makes the computation efficient and is expected to give real results.Future works are to include the circumnavigation problem involving a moving target.Asymptotic stability are to be established under some mild assumptions on the velocity of the target.

References

[1] Yongcan Cao (2015),"UAV circumnavigating an unknown target under a GPS-denied environment with range-only measurements" Addison-Wesley Professional.

[2] Zhiqiang Miao, Yaonan Wang and Rafael Fierro (2017), "Cooperative circumnavigation of a moving target with multiple nonholonomic robots using backstepping design".

[3] Cao, Y., Muse, J., Casbeer, D., Kingston, D. (2013)."Circumnavigation of an unknown target using UAVs with range and range rate measurements."

[4] Sastry, S., Bodson, M. (1989), "Adaptive control: Stability, convergence, and robustness".

[5] Hongwei Zhang, Hongwei Zhang, Zhihua Qu. Lyapunov, Adaptive, and Optimal Design Techniques for Cooperative Systems on Directed Communication Graphs, IEEE Trans.Automat Control 59(2012) 3026-3041. target using UAVs with range and range rate measurements."

[6] A brief introduction to ode45 in MATLAB. URL: www.eng.auburn.edu/ode45

[7] Euler method to solve ODE.URL: wiki/euler/method

Signature of Supervisor/s [illegible] 29/11/21

Signature of Students

Rashmi Mertia (B18EE040)

Arzoo Qureshi (B18EE063)