RADAR DECEPTION

THROUGH PHANTOM TRACK GENERATION

A Thesis

by

DIYOGU HENNADIGE ASANKA MAITHRIPALA

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

December 2005

Major Subject: Mechanical Engineering

RADAR DECEPTION

THROUGH PHANTOM TRACK GENERATION

A Thesis

by

DIYOGU HENNADIGE ASANKA MAITHRIPALA

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Suhada Jayasuriya
Committee Members,	Alexander Parlos
	Aniruddha Datta
Head of Department,	Dennis O'Neal

December 2005

Major Subject: Mechanical Engineering

ABSTRACT

Radar Deception

through Phantom Track Generation. (December 2005)

Diyogu Hennadige Asanka Maithripala, B.S., University of Peradeniya, Sri Lanka

Chair of Advisory Committee: Dr. Suhada Jayasuriya

This thesis presents a control algorithm to be used by a team of ECAVs (Electronic Combat Air Vehicle) to deceive a network of radars through the generation of a phantom track. Each ECAV has the electronic capability of intercepting and introducing an appropriate time delay to a transmitted pulse of a radar before transmitting it back to the radar, thereby deceiving the radar into seeing a phantom target at a range beyond that of the ECAV. A radar network correlates targets and target tracks to detect range delay based deception. A team of cooperating ECAVs, however, precisely plans their trajectories in a way all the radars in the radar network are deceived into seeing the same phantom. Since each radar in the network confirms the target track of the other, the phantom track is considered valid. An important feature of the algorithm achieving this is that it translates kinematic constraints on the ECAV dynamic system into constraints on the phantom point. The phantom track between two specified way points then evolves without violating any of the system constraints. The evolving phantom track in turn generates the actual controls on the ECAVs so that ECAVs have flyable trajectories. The algorithms give feasible but suboptimal solutions. The main objectives are algorithm development for phantom track generation through a team of cooperating ECAVs, development of the algorithms to be finite dimensional searches and determining necessary conditions for feasible solutions in the immediate horizon of the searches of the algorithm. Feasibility of the algorithm in deceiving a radar network through phantom track generation is demonstrated through simulation results.

ACKNOWLEDGMENTS

The following is my heartfelt appreciation to all those who gave me inspiration, advise, direction and insight to begin, conduct and complete this thesis.

Naturally, my greatest appreciation goes to my advisor, Prof. Suhada Jayasuriya for his guidance, support, motivation and patience without which, these pages would not have been written. I consider myself very fortunate for being able to work under his supervision.

With great pleasure, I extend my gratitude to both Dr. Alexander Parlos and Dr. Anirudda Datta, who willingly agreed to be on my thesis committee inspite of their busy schedules.

I would like to extend my deepest gratitude and appreciation to my parents, two bothers and my sister for their relentless support and unselfish love. I take this opportunity to thank all my friends here and back home for being an inspiration in all my work.

TABLE OF CONTENTS

Page

LIST OF FIGURES

FIGURE Page

CHAPTER I

INTRODUCTION

This study is motivated by a need for generating a so called "phantom track" by a set of co-operating ECAVs. In particular, the actions of each ECAV (Electronic Combat Air Vehicle) in a team are to be coordinated taking into consideration the physical limits of actuators or input constraints of each ECAV. The problem falls into the general class of coordinated path planning of a multi-agent system having nonholonomic constraints on each individual agent and tightly coupled inter-agent constraints. In this general setting we focus on the specific problem of radar deception through phantom track generation. The algorithms we present are essentially finite dimensional searches which reduce to one dimensional searches. This is motivated by the desire to make the algorithms computationally attractive and amenable to real time computations, thus ensuring practicality of implementation. The algorithms we develop have the potential to be generalized to be applicable to any multi agent cooperative system having nonholonomic individual agent constraints and inter agent constraints. However this final generalization along with proofs of existence of feasible solutions and existence of finite dimensional searches is beyond the scope of this thesis study.

A radar detects the presence of a target by listening into the echoes of its transmitted radio waves, bouncing off of the target. Measurements of the round-trip time and comparison of the frequency of the transmitted pulses to that of the moving target enables it to determine the range as well as the range-rate of the target. The radiation pattern of a radar will give rise to a main-lobe, where most of its radiated energy is concentrated, and much weaker side-lobes which are consequences of the cancelation and addition of radar waves. A target has to lie inside the main-lobe or a side-lobe to be identified by radar.

The journal model is *IEEE Transactions on Automatic Control.*

An ECAV is assumed to have the capability of intercepting and appropriately delaying the return of a transmitted pulse of a radar thereby making it see a phantom (false) target beyond the actual range of the ECAV. This capability of intercepting and digitally storing and returning encoded pulses is known as range delay. Range delay deception in its simplest form described above fails to deceive a radar network since radar stations in the network correlate the targets and target tracks amongst each other to identify range delay deception.

In view of this, a team of ECAVs cooperatively generate a single coherent phantom track thereby effectively deceiving the radar network. To achieve this, radar pulses are delayed by the ECAVs so that the perceived range vectors of each of the radars all intersect at a common point. The ECAVs are then repositioned so that they continuously stay in the line of sight of the radars at all subsequent times. This generates the phantom track of the desired speed and heading. Since each of the radars is deceived into seeing the same phantom track, the track is considered valid by the radar network.

The problem is formulated in two dimensions and thus the phantom point will have 2-DOF while each ECAV will have only a single DOF. The loss of a DOF for the ECAV is due to the constraint that it has to be inline with the phantom point and the radar that is assigned to it. Though the actual implementable controls are on the ECAVs, the approach to the problem is to formulate a phantom track that would guarantee existence of feasible controls on the ECAVs for generating this formulated phantom track.

The decentralized algorithms developed allows for computational savings with the additional requirement that communications between the ECAVs be kept to a minimum. Decentralization naturally requires that the algorithm not be computationally intense. The approach followed in phantom track generation presented here is unique in that the phantom track is not known a priori and is hence generated by the algorithm as part of its solution and this allows a lot of flexibility in the trajectory generation and the initial conditions of

the ECAVs.

A. Research Objectives

We strive to develop finite dimensional algorithms that have the potential to be generalized to be applicable to any multi agent cooperative system having nonholonomic individual agent constraints and inter agent constraints. The algorithms would give feasible but sub-optimal solutions. The main objectives of the proposed research are as follows,

- Algorithm development for phantom track generation through a team of cooperating ECAVs.
- Developing the algorithms to be finite dimensional searches.
- Determining necessary conditions for feasible solutions in the immediate horizon of the searches of the algorithm.

B. Thesis Outline

The work presented here is organized into seven chapters of which this introduction is Chapter-I. Chapter-II introduces radar fundamentals required for the understanding of the problem setting. In Chapter-III the general technique of radar deception through phantom track generation is introduced. Chapter-IV formulates the kinematics of a single ECAV engaging a single radar and goes onto give a brief description of previous work in this area. The proposed algorithm for phantom track generation is introduced in detail in Chapter-V. Chapter-VI presents three specific cases of its implementation along with simulation results of trajectories for each case. Chapter-VII concludes with a discussion of proposed future work and conclusions of the study.

CHAPTER II

RADAR FUNDAMENTALS

This chapter gives a brief introduction to radar fundamentals that would help in the understanding of the problem setting better. The discussion is mainly from the two texts *Radar Principles* by Peyton Z. Peebles [1] and *Introduction to Airborne Radar* by George Stimson [2].

The word *radar* is an acronym for *ra*dio *d*etection *a*nd *r*anging [1]. A radar detects a target by transmitting radio waves and listening for their echoes. To keep transmission from interfering with reception some radars transmit the radio waves in pulses and listens for the echoes in between and for this reason they are known as pulsed radars. The rate at which the pulses are transmitted is what is known as the pulse repetition frequency (PRF). Radars can be broadly catagorized as *continuous-wave* or *pulsed* according to their waveforms. In a continuous-wave type the radar transmits wave signals continuously usually with constant amplitude but can have frequency modulation (FM) or constant frequency. In a pulsed type the radar transmits wave signals in pulses for the afore mentioned reason and can be with or without frequency modulation [1].

When the transmission of a radar is pulsed the range of a target can be determined directly by measuring the time between the transmission of each pulse and reception of the echo from the target; a technique called *pulse delay ranging*. When the measured time delay(the round-trip time) is t_d the range R to the target is simply determined by eq.(2.1) where C is the speed of electromagnetic waves which is the speed of light. Pulse delay ranging is simple and can be extremely accurate and for this reason it is the most widely used method of range measurement [2].

$$R = \frac{C \cdot t_d}{2} \tag{2.1}$$

By concentrating the radiated energy into a narrow beam and by searching the search area with it, a radar can determine the direction of its target. The ability of a radar to resolve targets in azimuth and elevation is determined primarily by the azimuth and elevation beam width of this narrow beam. The direction of an isolated target can be determined to within a very small fraction of the beam width since the amplitude of the received echo varies symmetrically as the beam sweeps across a target [2].

The radio frequency of the echoes a radar receives from a moving object is shifted relative to the frequency of the transmitted radio waves in proportion to the range rate of the object; a phenomenon known as the *Doppler effect*. This difference or shift in frequency of the returned signal from a object in relative motion to that of the transmitted signal is what is termed the *Doppler frequency offset*. By sensing the doppler frequency a radar can separate and resolve returns received simultaneously from different targets and determine range rates of them [2].

A pulse doppler radar has the ability to detect doppler frequencies thus being coherent. It being coherent and largely digital gives it quantum improvements in performance and reliability. It can detect small targets at long ranges, and can track them either singly or as a group of targets, all this while continuing to search for more [2].

A radar that has the capability of following the movements of one or more targets while continuing to search for more can be termed a *tracking radar*. The beam of a tracking radar is repeatedly swept through a search scan to detect targets. Once detected the radar can automatically track the target. Its relative velocity is computed on the basis of either the periodic samples of its range and direction obtained during the search scan or the continuous data obtained by training the beam of the radar on the target [2].

An important characteristic of any radar is its radiation pattern. Radiation intensity patterns of a typical radar are given in Fig.1. There is usually a small region along a direction where the intensity is largest and this region is called the main lobe of the radar.

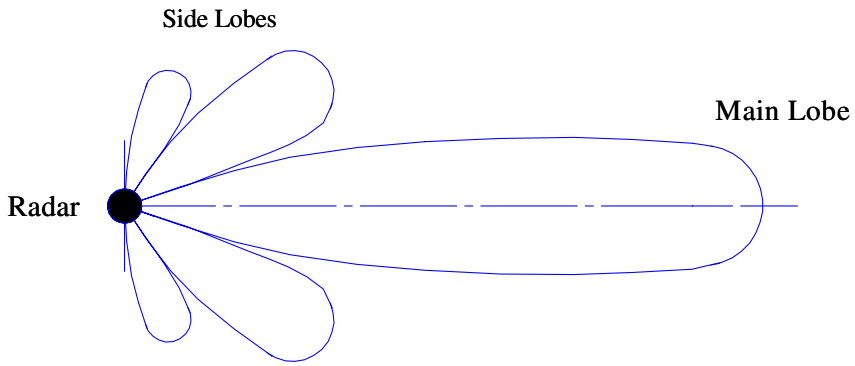

Fig. 1.: *Idealized radiation pattern of a typical radar*

A typical radiation intensity pattern has side lobes, in directions outside the main beam, that usually (and desirably) have maximums much smaller than that of the main lobe. In aggregate these side lobes rob the main lobe of a substantial amount of power. Side lobes are usually reduced through antenna design. Generally three characteristics of radiation pattern are of interest. Width of the main lobe, gain of the main lobe and the relative strengths of the side lobes. Antenna gain can be defined as the ratio of the power per unit of solid angle radiated in a specific direction to the power per unit of solid angle that would have been radiated had the same power been radiated uniformly in all directions. Another key radar characteristic is the rate or frequency of pulse transmission of a pulsed radar termed the pulse repetition frequency (PRF) [2].

Since transmitters are typically operated near their peak power limitation, many radars seek to transmit long-duration pulses to achieve high energy for good detection. For good range measurement accuracy on the other hand, a radar needs short pulses. By making use of the fact that the bandwidth of a long duration pulse can be made larger by frequency modulation, *pulse compression* provides a means to achieve both of these. With frequency modulation a waveform can be designed to have both small effective duration and long duration. Applying the long-duration waveform to its *matched filter* produces the waveform with small effective duration. The matched filter, also called the pulse compression

filter, has a constant modulus transfer function but a phase that corresponds to a linearly decreasing envelope delay. When the slope of this matches with the frequency modulation of the input signal, all the frequencies can be thought of as emerging at the same time and piling up in the output as illustrated in Fig.2. Side lobes are the unwanted by products of this pulse compression process [1].

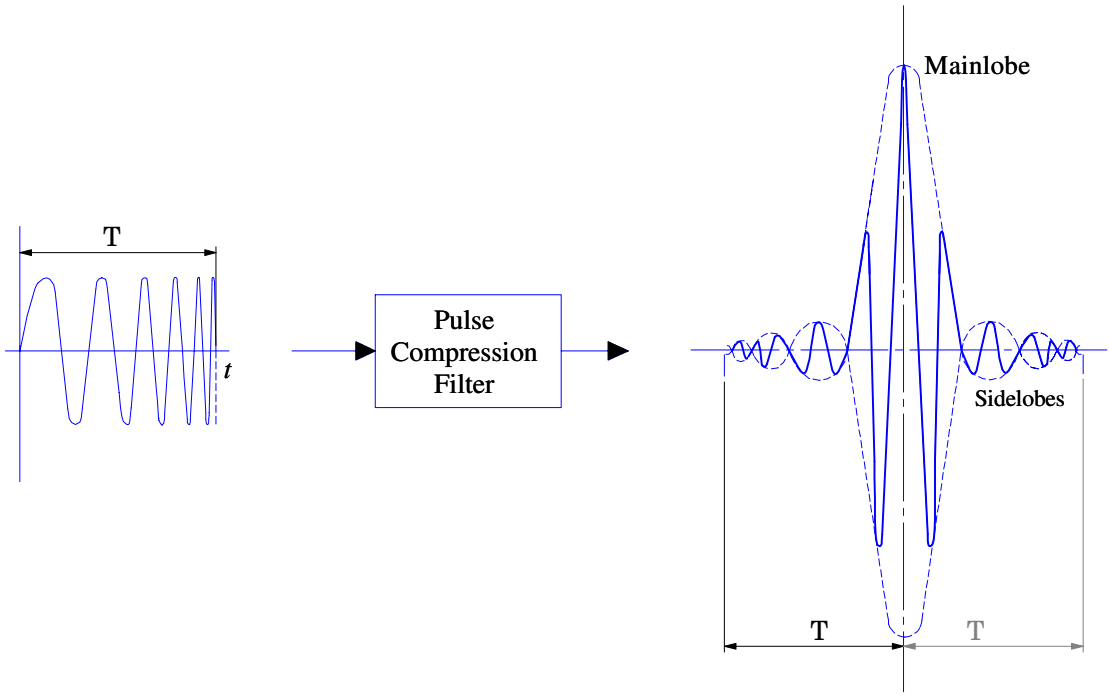

Fig. 2.: *Pulse compression of a radar*

Depending on its mode of operation, a radar can be *mono-static*, *bi-static* or *multi-static*. In mono-static radar the transmit and receiving stations exist at the same point while in a bi-static radar the transmit and receiving stations are at separate locations. A multi-static radar system has one or more transmitting stations and more than one receiving station all at different locations [1].

Integrated radar networks, for example a multi-static radar system integrated in a radar network, will have the ability to correlate target tracks within their operational ranges.

CHAPTER III

RADAR DECEPTION

This chapter introduces the reader to some concepts of Electronic Warfare(EW) and the general technique of radar deception through phantom track generation.

A. Introduction to Electronic Warfare

Radar deception falls into the field of Electronic Warfare and hence we give here a brief discussion of some of the terms and definitions used in this field for the benefit of the reader.

Electronic Warfare (EW) is defined as a military action involving the use of electromagnetic energy to determine, exploit, reduce or prevent hostile use of the electromagnetic spectrum and action which retains friendly use of the electromagnetic spectrum [3].

EW is organized into three major categories; Electronic warfare Support Measures (EMS), Electronic Counter Measures (ECM) and Electronic Counter Counter Measures (ECCM).

EMS is defined as the actions taken to search for, intercept, locate and immediately identify radiated electromagnetic energy for the purposes of immediate threat recognition and the tactical deployment of forces [3].

ECM is defined as actions taken to prevent or reduce the enemy's effective use of the electromagnetic spectrum. ECM includes jamming and deception. Jamming is the deliberate radiation or reflection of electromagnetic energy with the object of impairing the deployment of electronic devices, equipment or systems being used by hostile forces. Deception is the deliberate radiation, reradiation, alteration, absorption or reflection of electromagnetic energy in a manner intended to mislead a hostile force in the interpretation or use of information received by the electronic systems. Deception can be categorized as manipulative and imitative. Manipulative implies the alteration or simulation of friendly

electromagnetic signals to accomplish deception while imitative consists of introducing radiation into hostile channels which imitates a hostile emission. The radar deception considered in this study falls into the latter category. The most common form of ECM is noise jamming while pulse doppler radars are the least susceptible to noise jamming of all radar types [3].

ECCM is defined as actions taken to ensure friendly use of the electromagnetic spectrum against ECM [3].

An ECAV is assumed to have the stealth capability of not being detected by radar and the ECM capability of intercepting and appropriately delaying the return of a transmitted pulse of a radar, thereby making it see a phantom target at a range beyond that of the ECAV. This capability of intercepting, digitally storing and returning encoded returns is what is commonly known as *range gate deception* or *pull-off* or simply *range delay deception*. An ECAV achieves this by using an onboard DECM(Deceptive Electronic Counter Measure) system, often implemented in the form of a repeater jammer. Repeater DECM systems radiate replicas of the victim radar's signal, delayed in time, modulated in amplitude and shifted in Doppler frequency as is appropriate. The distinct characteristic of a repeater DECM system is that the victim signal is coherently stored in the repeater in digital memory making the returns more realistic.

B. Radar Deception through Phantom Track Generation

The range delay deception technique presented above can be effective against individual tracking radars but in general will fail against an integrated radar network. This is because of the ECCM capability of the integrated radar network to correlate target tracks within the network of radars to detect range delay deception. However an integrated radar network can be effectively deceived by a team of cooperating ECAVs through the generation of a

single coherent phantom track as will be explained through an example scenario. Figure 3 illustrates how a team of four ECAVs cooperate to generate a coherent phantom track through range delay deception to deceive an integrated radar network having four radar reception stations.

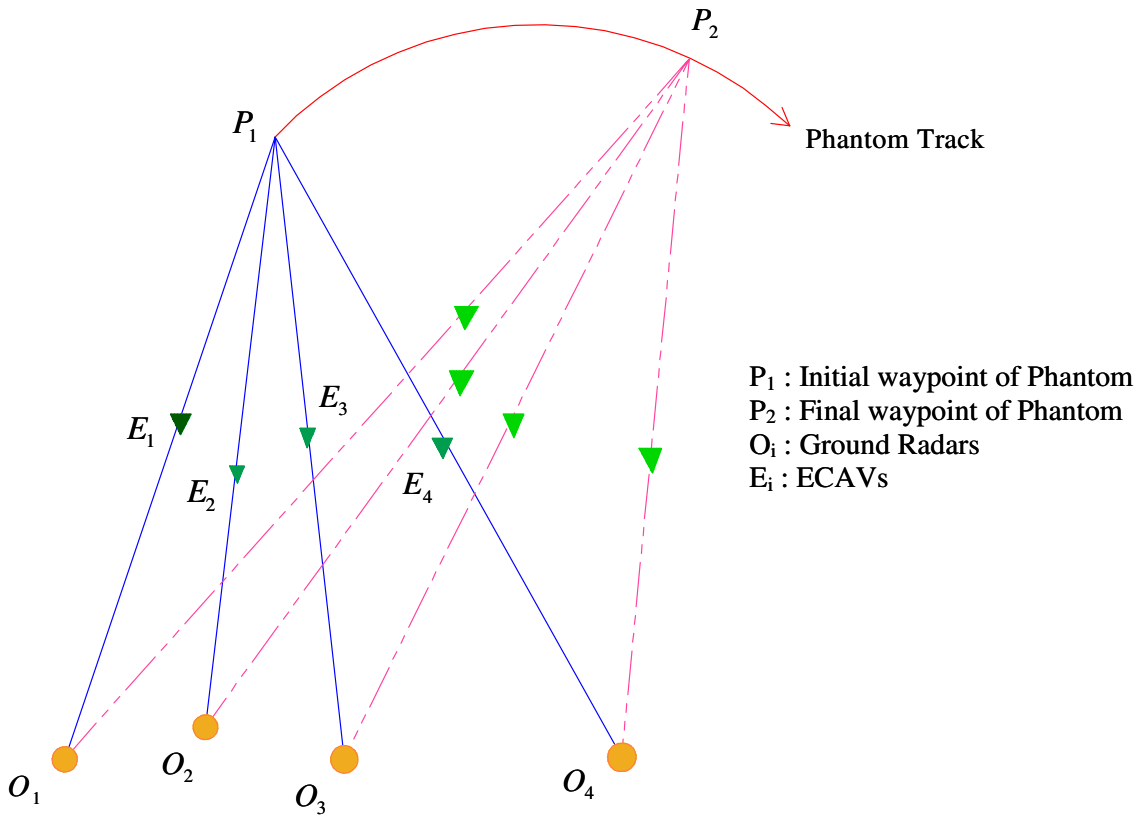

Fig. 3.: *Phantom track generation through a team of four ECAVs*

In this example scenario, there are four ECAVs, one assigned to each radar. The radars are assumed pulsed radars. At the start of the track, each ECAV is in the line of sight joining the corresponding radar location to the phantom position P_1. The radar pulses received by each ECAV are delayed appropriately so that the perceived range vectors all intersect at P_1. The ECAVs are then repositioned so that they continuously stay in the line of sight of the radar at all subsequent times. This generates the phantom track of the desired speed and heading at each time. Since each of the radars confirms the target track of the other, the

track is considered valid by the radar network.

The problem is formulated in two dimensions and hence the radars, the ECAV trajectories and the phantom track are all assumed to lie on a plane at a constant elevation.

CHAPTER IV

DYNAMICS OF PHANTOM TRACK GENERATION

This chapter presents the kinematics of a single ECAV engaging a single radar in section A and briefly discusses previous work carried out on this problem in section B.

A. Dynamics of Single ECAV, Single Radar Engagement

We start by formulating the kinematics of phantom track generation for the case of a single ECAV engaging a single radar station to keep the formulation as simple as possible. Figure 4 illustrates this simple case. The system dynamics here and in the rest of the thesis are restricted to the horizontal plane.

The ECAV and the radar states are (r, θ) and (R, θ) as shown in the figure. V and W are the instantaneous speeds of the phantom and the ECAV respectively and will in general be functions of time. The fundamental kinematic constraint of this dynamic system is that both the ECAV and the phantom will share the same bearing angle θ. The dynamic equations of the ECAV and the phantom are:

$$\pm\sqrt{\dot{r}^2 + (r\dot{\theta})^2} = \quad W \tag{4.1}$$

$$\pm\sqrt{\dot{R}^2 + (R\dot{\theta})^2} = \quad V \tag{4.2}$$

Realistic flight dynamics could be incorporated through constraints on the velocity, turn radius and acceleration. However this study does not consider acceleration bounds. Stall conditions and physical limitations on thrust of the ECAV will require that W be upper and lower bounded. Since the phantom attempts to mimic a real air vehicle, V too

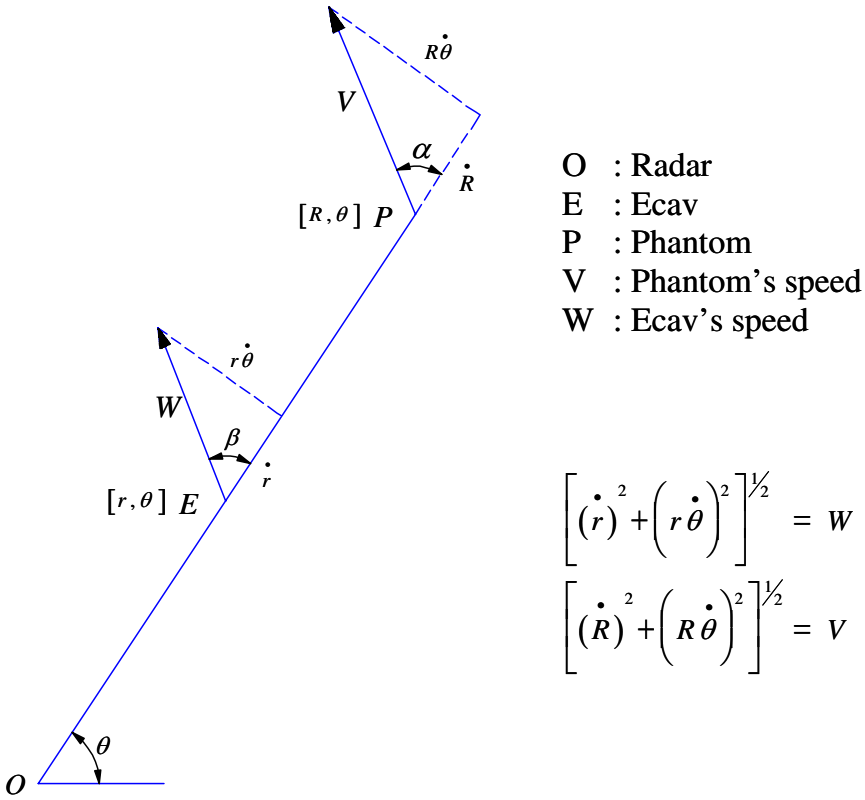

Fig. 4.: *ECAV and phantom track variables and their relationships*

will have upper and lower bounds on it.

$$W = [W_{min}, W_{max}] \quad where \quad W_{min}, W_{max} > 0 \tag{4.3}$$

$$V = [V_{min}, V_{max}] \quad where \quad V_{min}, V_{max} > 0 \tag{4.4}$$

Turn radius constraints are introduced through turn rate constraints on the ECAV and the phantom as given below.

$$\dot{\phi} = [-\dot{\phi}_{max}, \dot{\phi}_{max}] \quad where \quad \dot{\phi}_{max} > 0 \tag{4.5}$$

$$\dot{\psi} = [-\dot{\psi}_{max}, \dot{\psi}_{max}] \quad where \quad \dot{\psi}_{max} > 0 \tag{4.6}$$

The problem is essentially a constrained optimization leading to a classic two-point bound-

ary value problem(TPBVP).

When V is held constant, the dynamical equations given by eq.(4.1) and eq.(4.2) can be formulated in state space form as:

$$\dot{r} = W\cos\beta \tag{4.7}$$

$$\dot{\theta} = \frac{W}{r}\sin\beta \tag{4.8}$$

$$\dot{R} = [V^2 - W^2\frac{R^2}{r^2}\sin^2\beta]^{\frac{1}{2}} \tag{4.9}$$

where (r, R, θ) are the states, (W, β) are the controls and (R, θ) are the outputs.

Earlier research [4], [5] gives some interesting analytical solutions to the above dynamical system under certain assumptions and a short discourse of some of them is given below.

B. Previous Work

In [4] Pachter et. al., provides generalized mathematical formulations based on the kinematics of a single ECAV generating a phantom track against a single radar. The analytical and numenrical formulas that describe the trajectory of the ECAV is based on a *pre-determined* phantom track. Solving for the trajectory of the phantom, given a pre-determined ECAV trajectory is termed the *direct problem* while solving for the trajectory of the ECAV, given a pre-determined phantom trajectory is termed the *inverse problem*. The trajectory of the ECAV is solved for two specific pre-determined phantom tracks where the ECAV speed is assumed constant in each case. One where the speed of the phantom and heading is held constant resulting in a straight phantom track. The other where the trajectory of the phantom is circular with the speed of the phantom remaining constant. Next flyable regions for the ECAV are calculated by giving it flexibility in its speed and heading while ensuring that the pre-determined phantom track, be it straight or circular, is main-

tained. Incorporation of more flight dynamics through restrictions on minimum/maximum velocity, turn-radii, acceleration and extending the kinematics and mathematical formulas to the 3-D case are cited as future work.

In [5] Purvis et. al., shows that the assumption of constant speed ECAV and phantom track in [4] puts severe limitations on the valid initial conditions and trajectories of the ECAVs. Hence the ECAV and the phantom speeds are allowed to vary within certain bounds.

The phantom has 2-DOF, R and θ, while the ECAV has only one DOF, r, since it is constrained to stay inline with its radar and the phantom. For a single ECAV engaging a single radar, given a straight or circular phantom track, [5] solves for the ECAV trajectory by constraining the remaining single DOF of the ECAV in six different ways. That is by holding constant, the ECAV speed, course, heading, speed-rate, turn-rate and acceleration. Next flyable regions for the ECAV are calculated for pre-determined constant course(straight line) and circular phantom tracks. The flyable regions represent the union of all positions the ECAV can visit on different trajectories to create the given phantom track. The range of valid initial conditions for an ECAV to generate a straight or circular phantom track are also presented.

Next the decentralized cooperative control of a team of ECAVs generating a coherent straight or circular phantom track is formulated based on the concept of coordination functions, the flyable regions and bounds on the initial conditions of the ECAVs. Coordination functions are used in deciding upon a final phantom track from the set of all possible phantom tracks that all ECAVs can create by minimizing appropriate costs. Each ECAV passes its coordination function to the team leader which determines the team optimal phantom track by minimizing the total cost of the team.

In both [4] , [5] the phantom track is predetermined by hypothesis and only straight or circular phantom tracks are considered. Also given this pre-determined phantom track

the ECAV trajectory solution requires specifying one ECAV variable as a function of time, be it speed, course, heading, turn-rate, speed-rate or acceleration.

Both of these taken together severely restricts the freedom of the kinematics of the problem. These can be relaxed by making phantom track generation an integral part of the trajectory generation of the ECAV as apposed to assuming it pre-determined and by removing the requirement of specifying one ECAV variable as a function of time. Then the dynamical constraints of the ECAV are the ones coming from its flight dynamics and what ever additional constraints coming from the problem setting such as having to stay within a certain radius from the radar it engages. Of course, the flexibility gained in the feasible ECAV trajectory generation and the allowable initial conditions are at the expense of now having to solve the original TPBVP. As is well known solving a TPBVP is a difficult task at best and computationally intense. Consequently, it is not likely that one could generate solutions that can be implemented in real time and online. Instead in this study we propose algorithms that are computationally attractive and are amenable to real time computations. The algorithms we develop are essentially finite dimensional searches which can further be reduced to one dimensional parameter searches.

CHAPTER V

AN ALGORITHM FOR PHANTOM TRACK GENERATION

This chapter develops the proposed algorithm for phantom track generation. The algorithm is initially formulated for the case of a single ECAV engaging a single radar and the approach is then extended to the case of multiple ECAVs cooperating to deceive a network of radars. Control strategies to deal with different system constraint requirements are discussed.

A. Introduction

The formulation of the algorithm will be in discrete time and hence the trajectory generation will be through the generation of a finite set of waypoints. The following assumptions are made in the problem formulation where the phantom trajectory is not specified except for the initial and final waypoint. Also it is assumed that the velocities are determined by the constraints placed on the control inputs.

1. Assumptions

- Problem is restricted to the plane.
- ECAVs will have DECM (Deceptive Electronic Counter Measure) capabilities.
- Phantom trajectory not specified except for the initial and final waypoints.
- Both the phantom track as well as the ECAVs will have constrained dynamics.
- Phantom speed is greater than ECAV speeds.
- ECAVs are mass-less and their states are completely observable.

- All ECAVs are initially in-line with the radar it votes on and the initial waypoint of the phantom track.
- Ground radar locations are fixed and known to the ECAVs.
- Creation of the phantom track is through main-lobe deception using range delay techniques.

The basic approach followed to generate the phantom track through the trajectory generation of cooperating ECAVs is to incrementally generate a set of waypoints for the Phantom track that would guarantee existence of flyable trajectories for the ECAVs in each step of the incremental process. Sub-level controls would then be required to generate the final path connecting the intermediate waypoints for the ECAVs to fly.

Figure 5 illustrates how the algorithm would generate waypoints for the phantom and the i^{th} ECAV engaging its i^{th} radar. Current states of the ECAV and the phantom are such that they are collinear with the radar. f_E then maps the current state of the ECAVs at time t to the set of all possible states, $f_E(E_i)$, the ECAV can occupy at time $t + \Delta t$. Here Δt is the incremental time step of the algorithm, where at every Δt time the algorithm generates waypoints for the ECAV and the phantom.

The mapping f_E represents the flight dynamics of the ECAVs restricted to the plane along with any other additional constraints that we may want to impose on the ECAV trajectories. One desirable constraint on the ECAV trajectory would be for it to not get too close to its radar to avoid detection. Likewise f_P maps the current state of the phantom at time t to the set of all possible states, $f_P(P)$, the phantom can occupy at time $t + \Delta t$. The mapping f_P represents the flight dynamics of an actual or imaginary air vehicle it attempts to mimic, electronic delay dynamics of range delay and other constraints we may want to impose on its trajectory.

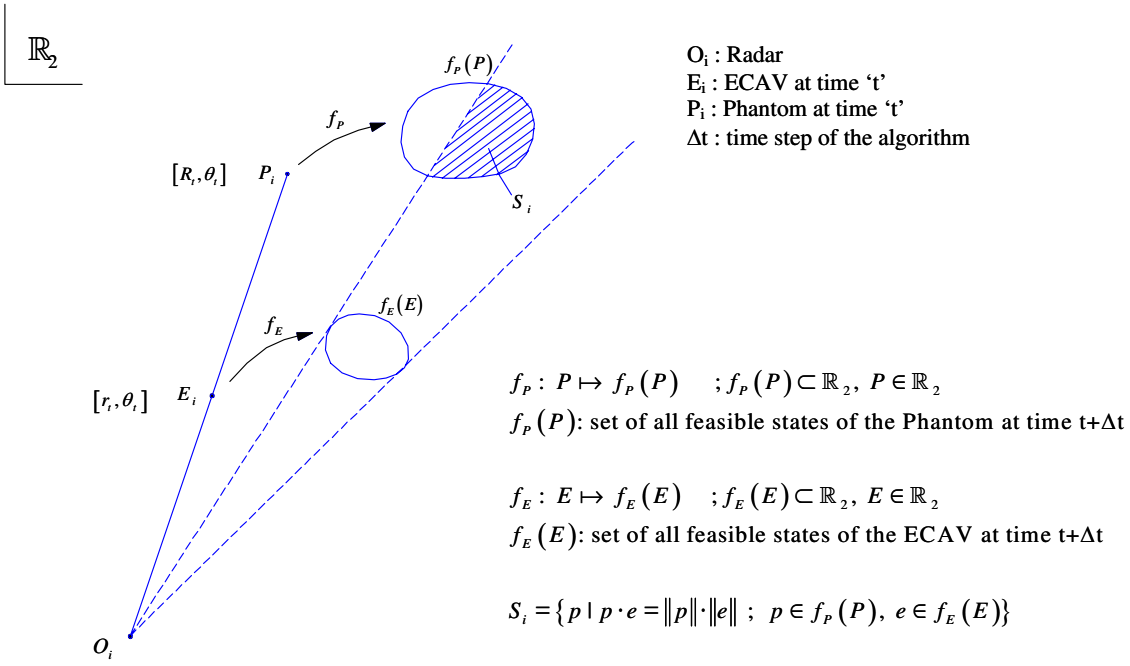

Fig. 5.: *Illustration of the basic approach to the algorithm*

Starting from current states at time t where the ECAV, phantom and the radar are collinear, waypoints for the ECAV and the phantom must be chosen from the sets $f_E(E_i)$ and $f_P(P)$ such that at the lapse of time Δt the ECAV, phantom and the radar will once more be collinear. The approach followed by the algorithm to achieve this is to first determine a waypoint for the phantom that would guarantee existence of a feasible waypoint for the ECAV to pick from without violating the collinearity constraint. This approach is in place to make the algorithm scalable to n number of ECAVs engaging n number of radars as will be clear from the discussions to follow. Once the sets $f_E(E_i)$ and $f_P(P)$ are constructed the next step of the algorithm is determining the subset S_i of $f_P(P)$ such that no matter what waypoint is picked for the phantom from this subset S_i, there will be at least one waypoint the ECAV can pick from that would not violate the collinearity constraint. That is to say

$S_i \subset f_P(P)$ is

$$S_i = \{p \mid p \cdot e = \|p\|\|e\| \quad ; \quad p \in f_P(P), e \in f_E(E_i)\} \tag{5.1}$$

Here $\|\cdot\|$ is the Euclidean norm. If $S_i \neq 0$, the next waypoint $s \in S_i$ for the phantom could be determined based on minimizing some cost, an example being a cost associating the distance to the final desired waypoint of the phantom track. Obviously if $S_i = 0$ we cannot proceed.

Once a waypoint $s \in S_i$ for the phantom is determined the subset Q_i of $f_E(E_i)$ that satisfies the collinearity constraint with this fixed s is determined. In mathematical terms Q_i is

$$Q_i = \{q \mid q \cdot s = \|q\|\|s\| \quad ; \quad q \in f_E(E_i)\} \tag{5.2}$$

The waypoint $q_i \in Q_i$ for the ECAV is chosen, again based on minimizing an appropriate cost. Once q_i is determined, controls are applied that would take the ECAV from its current state E_i at time t to q_i at time $t + \Delta t$. The nature of the two trajectories of the ECAV and the phantom thus depend on the costs associated with finding $s \in S_i$ and $q_i \in Q_i$.

We now extend this approach to the case of two ECAVs engaging two radars to generate a single coherent phantom track as illustrated in Fig.6. S_1 is the subset of $f_P(P)$ that ensures the existence of feasible waypoints for the first ECAV to pick from while $S_2 \subset f_P(P)$ ensures the existence of feasible waypoints for the second ECAV. Feasible here refers to the feasibility of ensuring the collinearity of the phantom, the ECAV and the radar it engages.

The intersection of these two sets, $S = S_1 \cap S_2$, then ensures the simultaneous existence of feasible waypoints for both the ECAVs. If S is non empty, $s \in S$ is determined that minimizes an appropriate cost and based on this s, the two subsets Q_1 and Q_2 are

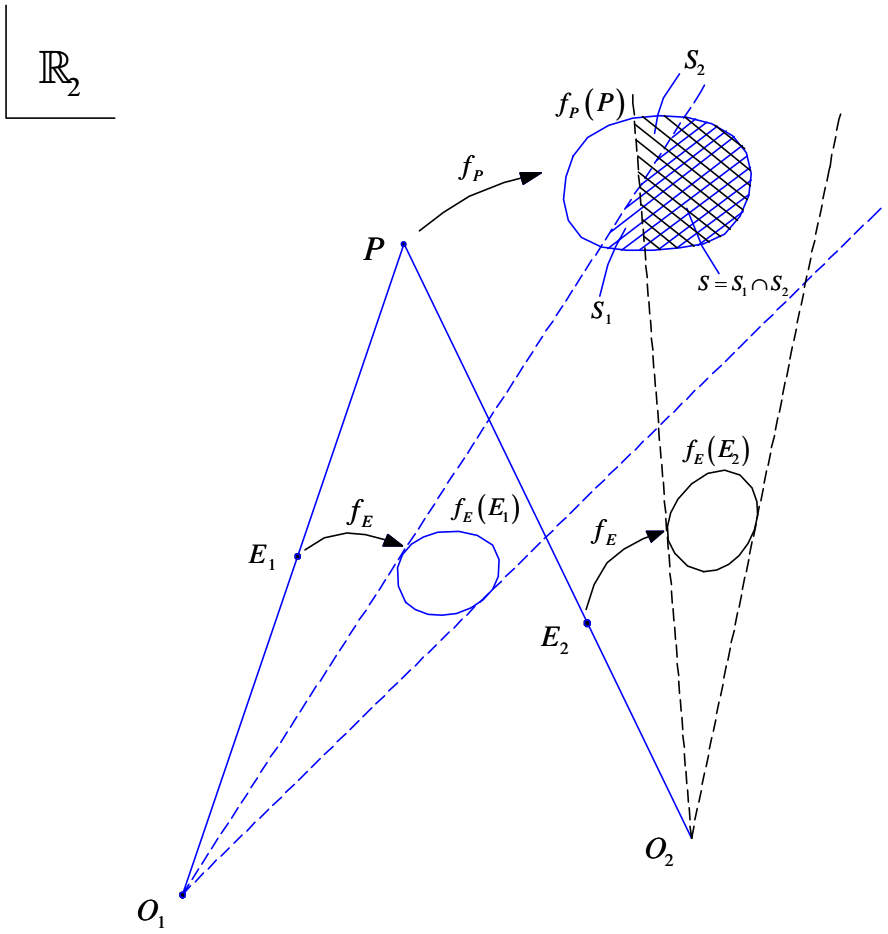

Fig. 6.: *Illustration of the basic approach to the algorithm for the case of two ECAVs engaging two radars*

determined such that;

$$Q_1 = \{q \mid q \cdot s = \|q\|\|s\| \quad ; \quad q \in f_E(E_1)\}$$

$$Q_2 = \{q \mid q \cdot s = \|q\|\|s\| \quad ; \quad q \in f_E(E_2)\}$$

Waypoints of the two ECAVs, $q_1 \in Q_1$ and $q_2 \in Q_2$, are also determined by minimizing appropriate costs. By incrementally stepping forward in time the algorithm would generate waypoints for the phantom and ECAV trajectories as long as S is non empty. The number of waypoints generated would also be a function of the incremental time step Δt of the algorithm.

One interesting possibility would be to associate the costs of finding q_1 and q_2 with high penalties for the set $S = S_1 \cap S_2$ becoming empty in the iterations to follow. By minimizing this particular cost the algorithm would run in a direction that would generate a coherent phantom track for the longest possible time.

It should be clear that the approach given above easily extends to n number of ECAVs engaging n number of radars to create a single coherent phantom track. In an autonomous team of n-ECAVs, each ECAV determines its corresponding set $S_i \subset f_P(P)$ and passes it onto the team leader. The team leader then determines the intersection $S = S_1 \cap ... \cap S_n$ of the n received sets and picks a waypoint $s \in S$ for the phantom. Once $s \in S$ is chosen it is passed onto all n-ECAVs where each of them, based on this information, determines its own waypoint.

B. Control Strategies

The algorithm presented above has three design degrees of freedom, the freedom in selecting $s \in S$, $q_i \in Q_i$ and Δt. The way these are determined will determine the nature and success of the trajectory generation.

Maintaining $S \neq 0$ is imperative for the successful implementation of the algorithm and hence it is necessary to associate the two costs determining $q_i \in Q_i$ and $s \in S$ with high penalties for the set $S = S_1 \cap ... \cap S_n$ falling empty in iterations to follow.

Since we assume the problem is restricted to the $2 - D$ plane, collision avoidance must be accounted for explicitly. From a control point of view this can be achieved by associating the cost that picks $q_i \in Q_i$ with a large penalty for any two ECAVs getting too close to each other. On the other hand maintaining communication connectivity amongst the ECAVs of the team constraints the maximum allowable distance between them. Hence it pays to associate the cost that determines $q_i \in Q_i$ with a high penalty for ECAVs moving

too far away from each other. Issues of collision avoidance and maintaining communication network connectivity in a team of cooperating ECAVs is addressed in [6].

In a realistic radar deception scenario it would be desirable to have the ECAVs always be sufficiently far away from the radar stations to minimize the threat of being detected through visual observation. This could be achieved by associating the cost that picks $q_i \in Q_i$ with a high penalty of getting too close to any of the radar stations in its radar network.

All radar have maximum detection ranges depending on their power capacities and hence it would be desirable to restrict the phantom track to lie within the radar space of the radar network. Again this can be achieved through the use of an appropriate cost for choosing $s \in S_i$ that penalizes the phantom moving out of the radar space. This obviously requires the ECAVs have prior knowledge of the maximum detection range of each of the radars they engage.

It is clear that the costs determining $s \in S$ and $q_i \in Q_i$ each have to contend with multiple system constraints. When a single cost has to contend with multiple system constraints the total cost can be constructed as a sum of weighted costs each corresponding to a system constraint. One strategy would be to have dynamic weighting where system constraints that become critical are assigned larger weighting in the total cost. A simpler strategy would be to assign constant weights based on a system constraint hierarchy.

CHAPTER VI

APPLICATION OF THE BASIC ALGORITHM TO THREE DIFFERENT CASES

Having formulated the general algorithm its implementation on three specific cases arising from different system constraints are addressed below.

A. Bounds on Rates

For this case we constrain the system dynamics through bounds on the range rate and angular rate of the ECAVs and the Phantom. We let the phantom state be (R, θ), ECAV state be (r, θ) and the constant bounds on the rates be

$$\dot{r}_{min} \leq \dot{r} \leq \dot{r}_{max} \tag{6.1}$$

$$\dot{R}_{min} \leq \dot{R} \leq \dot{R}_{max} \tag{6.2}$$

$$\omega_{min} \leq \dot{\theta} \leq \omega_{max} \tag{6.3}$$

These constraints lead to $f_P(P)$ and $f_E(E)$ being rectangular regions for the phantom and the ECAV as shown in Fig.7. The assumption we make that the ECAVs are mass-less allows us to look at $f_E(E)$ as the feasible velocity sector of the ECAV at E when Δt is of unit time. This is because under this assumption the ECAV can instantaneously change its velocity at state E. Similarly $f_P(P)$ is then the feasible velocity sector of the phantom at state P. The bounds on the rates of the state of the ECAV will be functions of the flight-dynamics of the ECAV as well as of its state. The two bounds on the range rate of the phantom will be functions of the flight-dynamics of the ECAV, the state of the ECAV and the dynamics of range delay.

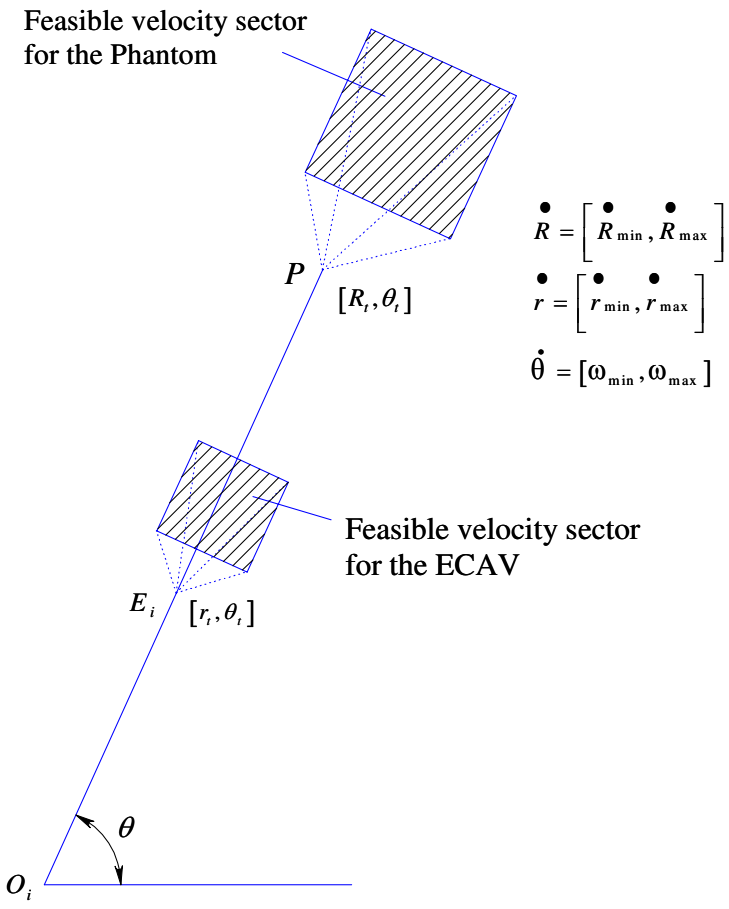

Fig. 7.: *Feasible velocity sectors for the ECAV and the phantom for bounded rates*

1. Simulation results for the case of bounds on rates

In the absence of adequate knowledge on these bounds, constant bounds are assumed on each of the rates. For the ECAV and the current states of the phantom to be contained within their respective velocity rectangles it is clear that each rate has to have bounds of opposite sign. For a set of constant bounds that would result in the phantom not being contained within its velocity rectangle, for any given set of initial conditions there will always be a region the phantom can never reach or enter. The simulation result of the algorithm implemented in MATLAB, which considers two ECAVs engaging two radars, given in Fig.8 is for such a set of constant bounds which does not result in the phantom

being contained within its velocity rectangle. As is seen from Fig.8, for the given set of

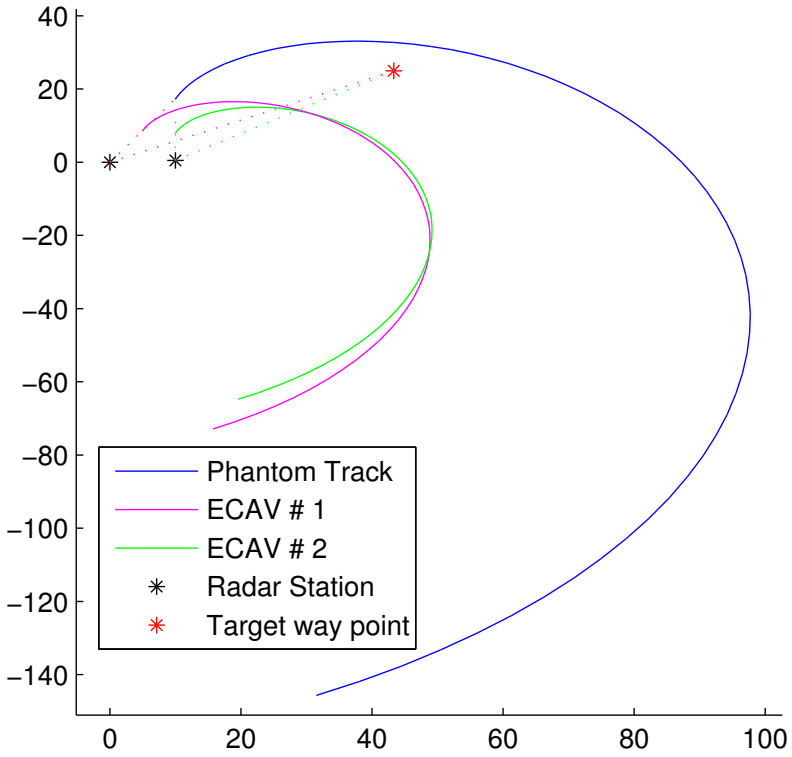

Fig. 8.: *Simulation results for phantom track generation using two ECAVs for the case of bounds on rates*

initial conditions the target waypoint of the phantom is in that region the phantom can never reach. It is clear that the time varying nature of each of the bounds is critical for this particular approach to constraining the system dynamics.

B. Bounds on Ground Speeds

Here we consider system constraints placed through upper and lower bounds on the ground speeds of the ECAV and the phantom. Such constraints lead to $f_P(P)$ and $f_E(E)$ being annular regions as shown in Fig.9. Once again these two sectors can be treated as velocity

sectors of the ECAV and phantom, when the incremental time step Δt of the algorithm is of unit time. Considered in the figure is the case of a single ECAV engaging a single radar.

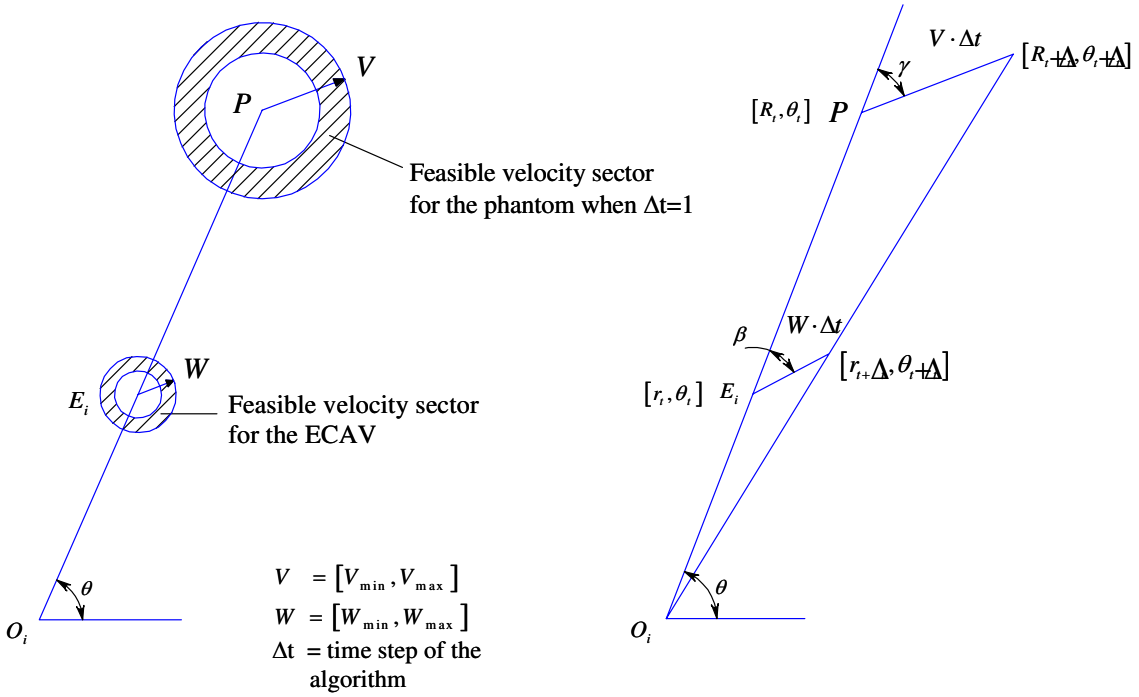

Fig. 9.: *Feasible velocity sectors of the ECAV and the phantom for bounded ground speeds*

As explained earlier the algorithm constrains the ECAV, the radar it engages and the phantom be collinear only at the end of each time step of the algorithm at which point the direction and speed are once more set for both the phantom and ECAV and maintained constant for the duration of the time step of the algorithm.

The collinearity constraint requires

$$\sin\beta = \frac{r_{t+\Delta t}}{R_{t+\Delta t}} \frac{V}{W} \sin\gamma \tag{6.4}$$

where V and W are the ground speeds of the phantom and the ECAV respectively. In each incremental step of the algorithm, the current states of the phantom and the ECAV

are given by (R_t, θ_t) and (r_t, θ_t) respectively while $(R_{t+\Delta t}, \theta_{t+\Delta t})$ and $(r_{t+\Delta t}, \theta_{t+\Delta t})$ gives their states at the end of each step of the algorithm. The angles γ and β are the angles defined in Fig.9.

A necessary condition for solutions for eq.(6.4) to exist is

$$\frac{r_{t+\Delta t}}{R_{t+\Delta t}} \frac{V}{W} \sin\gamma \leq 1 \tag{6.5}$$

Existence of solutions simply means that for a given waypoint of the phantom there will exist at least one feasible waypoint for the ECAV to pick from without violating the collinearity constraint. The condition for the existence of solutions is thus translated as a constraint on the feasible velocity sector of the phantom and this can be extended to the case of multiple ECAVs.

The hatched area of the annular region, which was the initial velocity sector of the phantom, in Fig.10 represents the feasible velocity sector of the phantom restricted through the necessary condition and is the set S_i introduced in eq.(5.1).

A sufficient condition for the existence of solutions to eq.(6.4) is

$$\gamma \leq \gamma_{cr} \tag{6.6}$$

where γ_{cr} satisfies,

$$\tan[\arcsin(\frac{W\Delta t}{r_t})] = \frac{V\Delta t \sin\gamma_{cr}}{R_t + V\Delta t \cos\gamma_{cr}}$$

Figure 11 shows three instances of a single ECAV engaging a single radar with the ECAV differently placed on the line joining the phantom to the radar. The areas of the annulus of the phantom, hatched in *black* corresponds to the necessary condition given by eq.(6.4) while the areas hatched in *red* corresponds to the sufficient condition given by eq.(6.6). As should be expected, the area corresponding to the sufficient condition is a subset of the area , S_i, the area corresponding to the necessary condition.

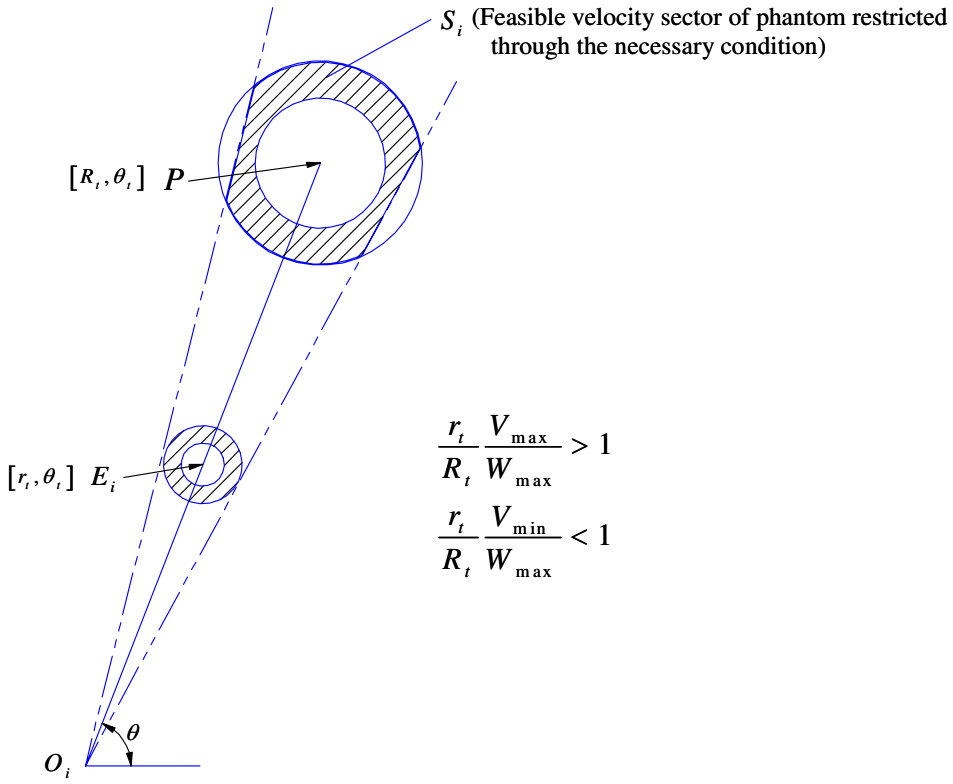

Fig. 10.: *Geometric representation of the necessary condition*

The third case of Fig.11, where $\frac{r_t}{R_t}\frac{V_{min}}{W_{max}} \leq 1$, shows that if the ECAV is sufficiently close to its radar then it does not impose any restriction on the sector annulus of the phantom and hence $S_i = f_P(P)$.

The algorithm whose results we present later on, implements the sufficient condition in place of the more difficult to implement necessary condition. It should be noted here that practical values of the bounds on the speeds makes these annuli narrow compared to their radii and hence the implementation of the sufficient condition in place of the necessary condition does not reduce the freedom allowed for the phantom. Hence with a little abuse of the mathematical notation introduced in the previous chapter, the set $S_i \subset f_P(P)$ for each ECAV is now found through the sufficient condition given in eq.(6.6) instead of through the correct necessary condition given in eq.(6.4). The feasible velocity sector S of the

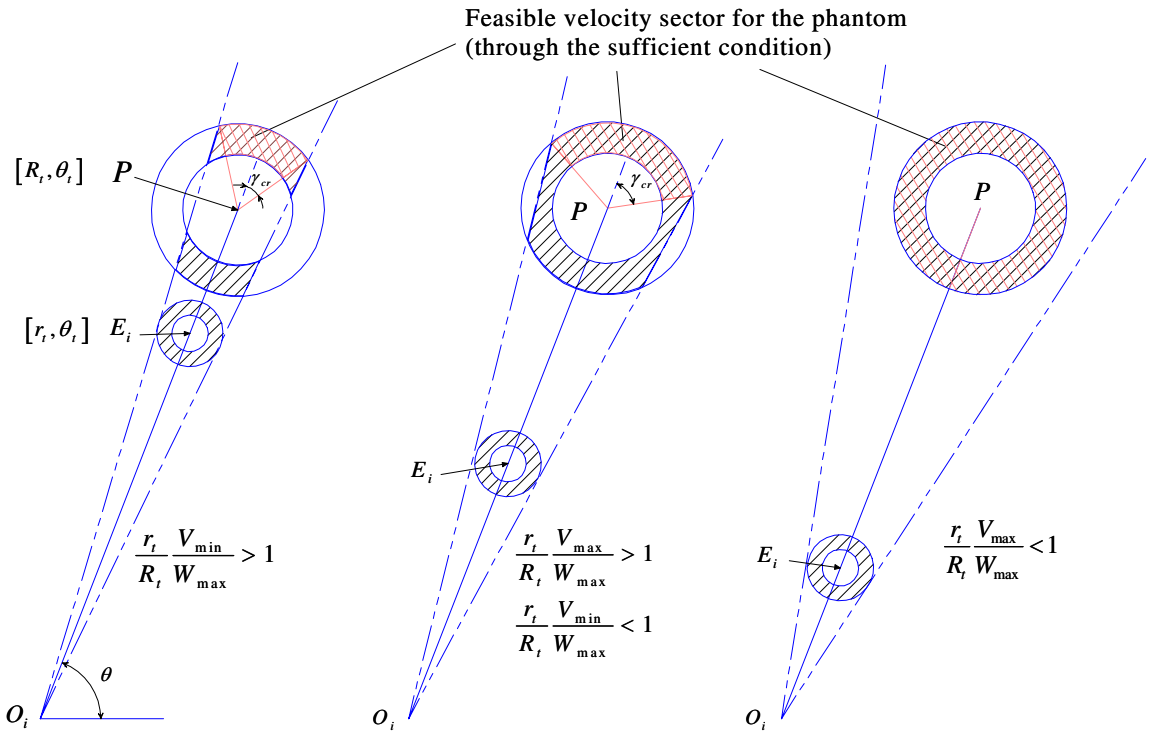

Fig. 11.: *Geometric representation of necessary and sufficient conditions*

phantom which would ensure the existence of solutions for each and every ECAV is then the intersection of these feasible velocity sectors S_i corresponding to each of the ECAVs. How the algorithm generates this intersection of the feasible velocity sectors S_i is best illustrated through an example where two ECAVs are engaging two radars. Considered in Fig.12 is the case where both the ECAVs impose its own restriction on the annular velocity sector of the phantom, through the sufficient condition. That is where, $S_1 \subset f_P(P)$ and $S_2 \subset f_P(P)$. The four points(marked with crosses in Fig.12) along with the boundary of the annulus $f_P(P)$ completely defines the two sets S_1 and S_2. Note that the edge points of $S = S_1 \cap S_2$ is a subset of the sector edges of S_1 and S_2 and thus S can be easily determined.

It is worth noting that in this approach to find the set S each ECAV needs to communicate only two pieces of information, namely the angle γ_{cr} and its orientation.

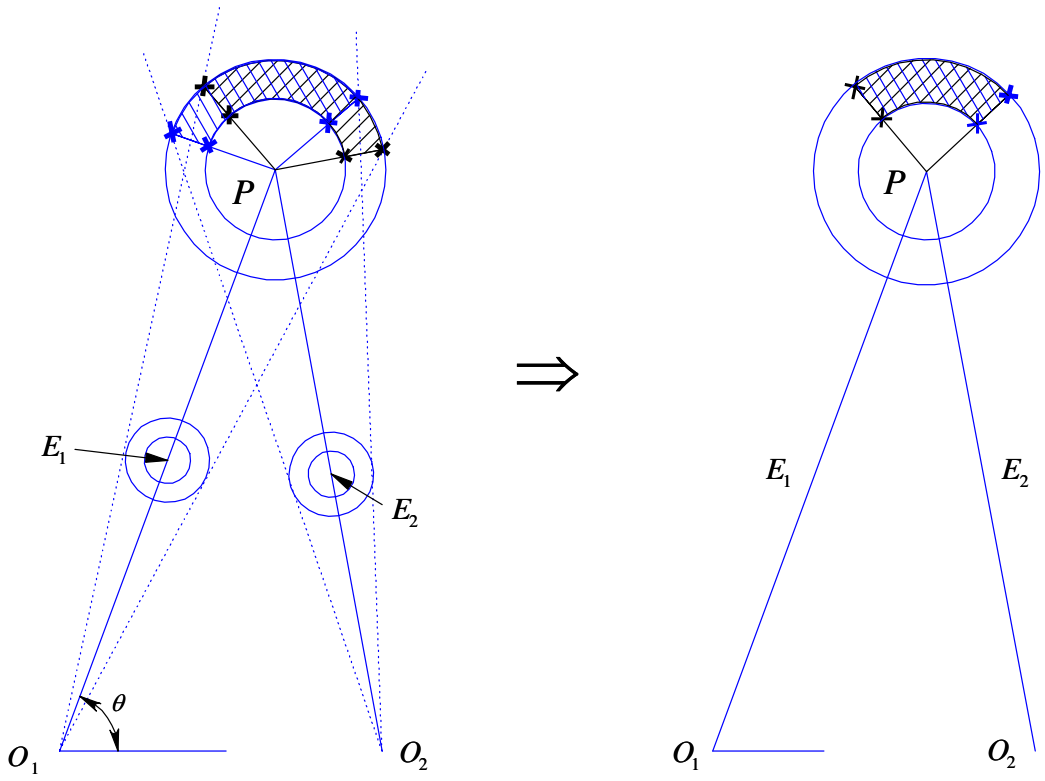

Fig. 12.: *Case of two ECAVs engaging two radars*

Next the algorithm would pick a velocity for the phantom from its final feasible velocity sector S which would make it travel towards the final desired waypoint in minimum time. This is done through a $2-D$ parameter search within S, since the set S is in the plane, by finding the point in S that is closest to the final desired waypoint of the phantom track. However searching along the boundary of S is sufficient except when the final waypoint falls within S reducing it to a $1-D$ parameter search.

If the states of any one of the ECAVs is such that it imposes a restriction on the movement of the phantom through the sufficient condition, it is interesting to note that no matter what velocity heading is picked for the phantom, the ECAV can travel only in a direction that would relax the restriction placed by it at the phantom. Figure 13 illustrates this point. Here $r_t V_{max} > R_t W_{max}$, which implies $\alpha < \beta$ for all $\alpha \leq \gamma_{cr}$ where the angle

α is defined in Fig.13.

It can be then shown that,

$$\frac{r_{t+\Delta t}}{R_{t+\Delta t}} < \frac{r_t}{R_t} \tag{6.7}$$

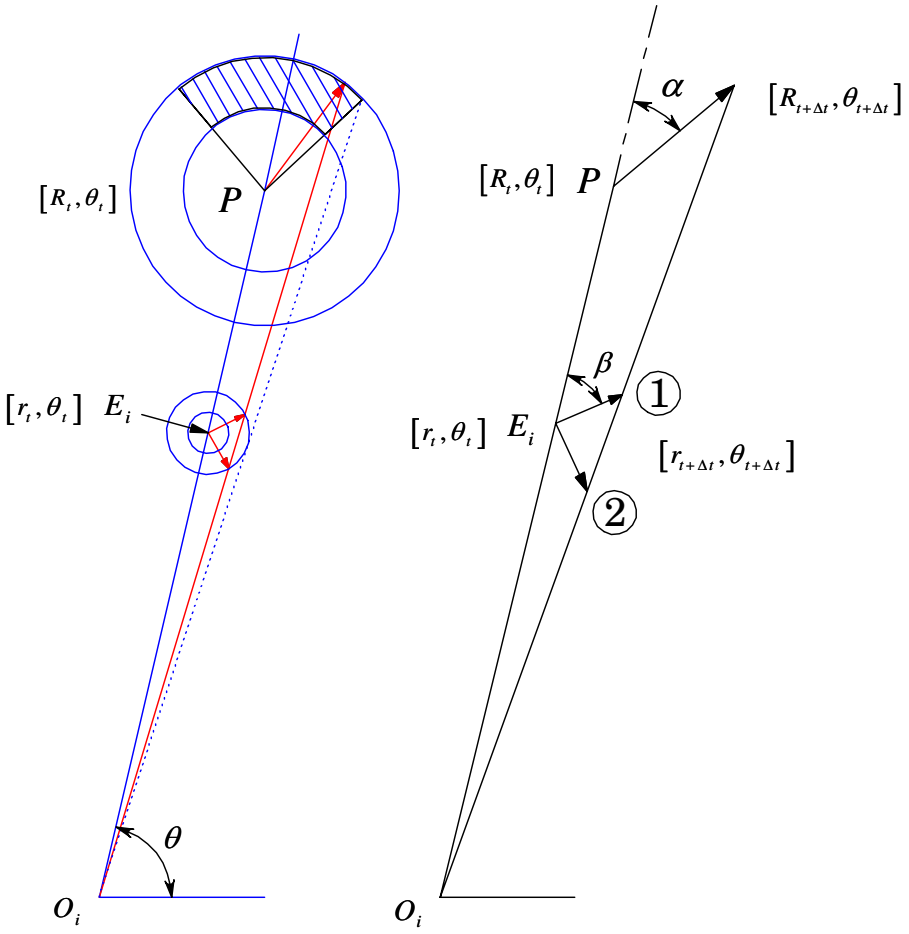

Fig. 13.: *ECAV velocities when the ECAV imposes a restriction on the velocity annulus of the phantom.*

Hence in the absence of additional constraints, any restriction placed by ECAVs on the velocity annulus of the phantom only relaxes with travel time and ultimately would cease to apply. That is to say,

$$Lim_{t\to\infty} S \to f_P(P)$$

It can be shown that the condition for an ECAV, the radar it engages and the phantom

to be collinear for the entire duration and not just at the end of the incremental time step of the algorithm is,

$$\frac{W_{min}}{V_{max}} \leq \frac{r_t}{R_t} \leq \frac{W_{max}}{V_{min}} \tag{6.8}$$

Since the algorithm is formulated in discrete time the instantaneous tangential and radial speeds are held constant and this couples the otherwise independent radial speed components as shown in Fig.14. The kinematics shown in Fig.14 gives us the relation,

$$\frac{r_t}{R_t} = \frac{W}{V}$$

This along with the bounds on W and V is what gives us the relation given above in eqn.(6.8).

In Fig.13, the set Q_i introduced in eq.(5.2) as the set of points the ECAV can pick from for its waypoint once a waypoint is picked for the phantom, will be a portion or the whole of the line segment between the circled numbers of one and two. Once a velocity(and hence a waypoint) for the phantom is picked, the algorithm picks a velocity for the ECAV from this set Q_i that takes the ECAV into or closest to the range given in eqn.(6.8).

The initial conditions all ECAVs should satisfy to guarantee a straight line path for the phantom from its initial waypoint to the final waypoint is,

$$\frac{r_{0,i}}{R_{0,i}} \leq \frac{W_{max}}{V_{min}} \tag{6.9}$$

where the *subscript* 0 denotes initial conditions while i indexes the ECAVs. With these initial conditions none of the ECAVs impose restrictions on the annular velocity region of the phantom arising from its speed bounds, and hence the algorithm generates a straight line phantom track connecting the initial to the final desired waypoint.

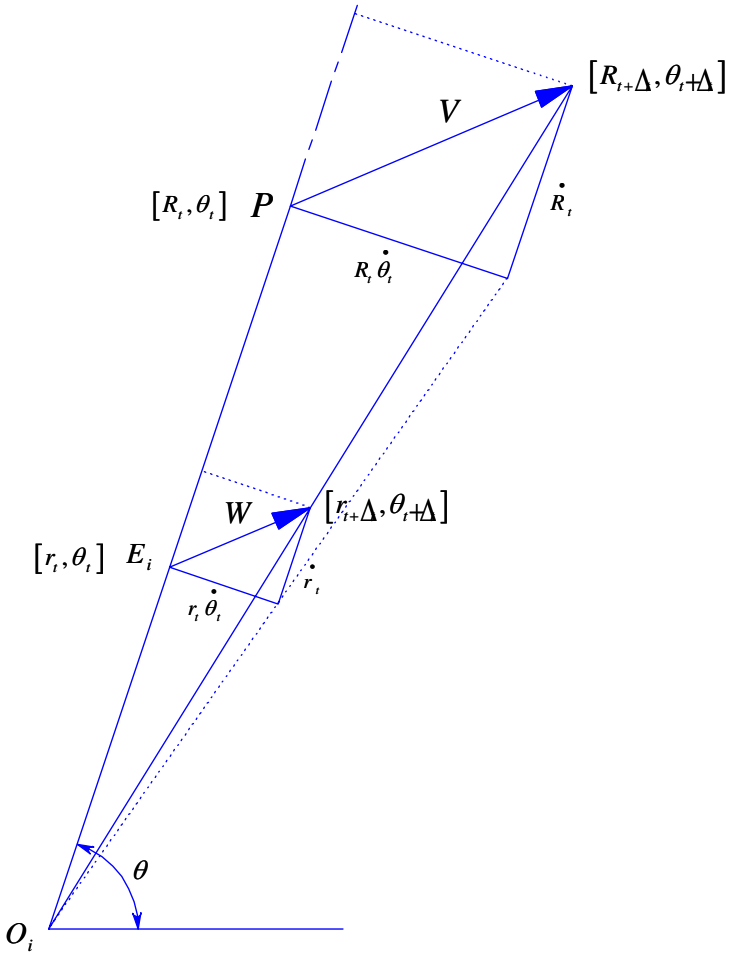

Fig. 14.: *Kinematics of maintaining collinearity for all times*

1. Simulation results for the case of bounds on ground speed

The algorithm was coded in MATLAB and Fig.15 gives the simulation results of the trajectories generated for the ECAVs and the phantom for the case of four ECAVs when the initial conditions are such that initial restrictions are imposed on the annular velocity sector of the phantom, which is to say initially $S \subset f_P(P)$. The phantom trajectory is generated from left to right. Simulation results are for a phantom speed of $400 \pm 40m/s$, ECAV speeds of $100 \pm 10m/s$ and an incremental step time of one second. The phantom track makes a sudden change in its course and starts traveling along a straight line towards the

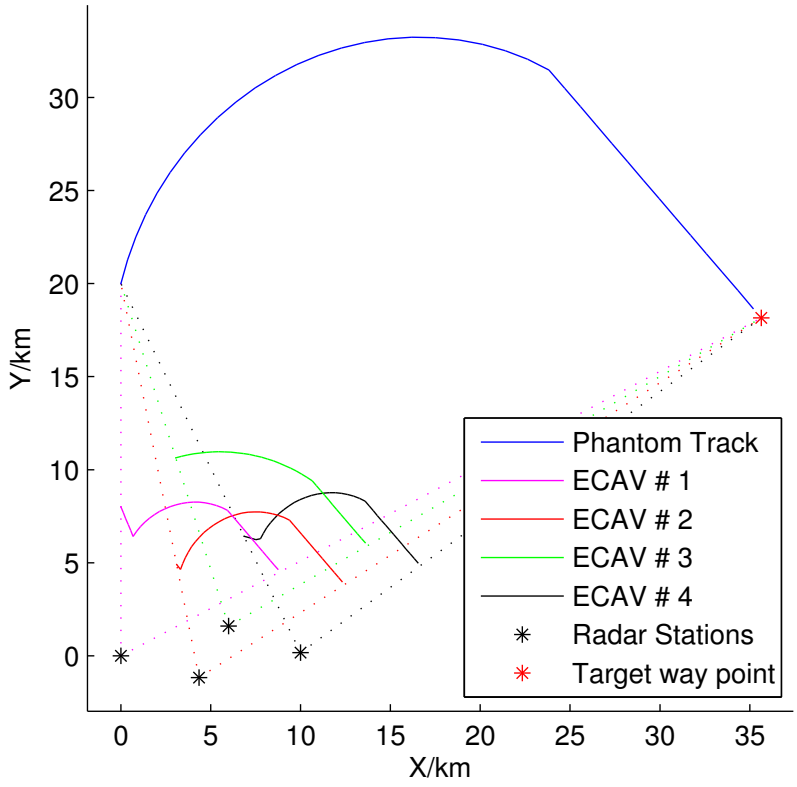

Fig. 15.: *Simulation results for a team of four ECAVs when initial conditions impose restrictions at the phantom for the case of bounds on ground speeds*

final waypoint when all restrictions placed on its annular velocity sector relaxes with time resulting in $S = f_P(P)$. As would be evident by a closer look at Fig.15 the phantom locking onto a straight line path corresponds with all ECAVs also locking onto straight line paths. This occurs due to the fact that the ECAV are treated as holonomic agents where each of them are free to make sudden directional changes.

We emphasize that modeling the flight dynamics as we did through simple constraints on rates or ground speed does not accurately model realistic flight situations, but at the same time they are necessary steps towards a more realistic dynamic setting. Additionally imposing turn rate constraints on the dynamic setting adopted here simulates a more realistic dynamic setting as will be shown by the results of the next section.

C. Bounds on Ground Speeds and Turn Rates

Next we consider the introduction of bounds on the turn rates in addition to the bounds on the ground speeds. The addition of the turn rate constraints leads to $f_P(P)$ and $f_E(E)$ being subsets of the annuli resulting from the bounds on the ground speed, as illustrated in Fig.16. As in the two earlier cases, the appropriate assumptions allows $f_P(P)$ and $f_E(E)$ to be viewed as feasible velocity sectors available for the phantom and the ECAV at time t.

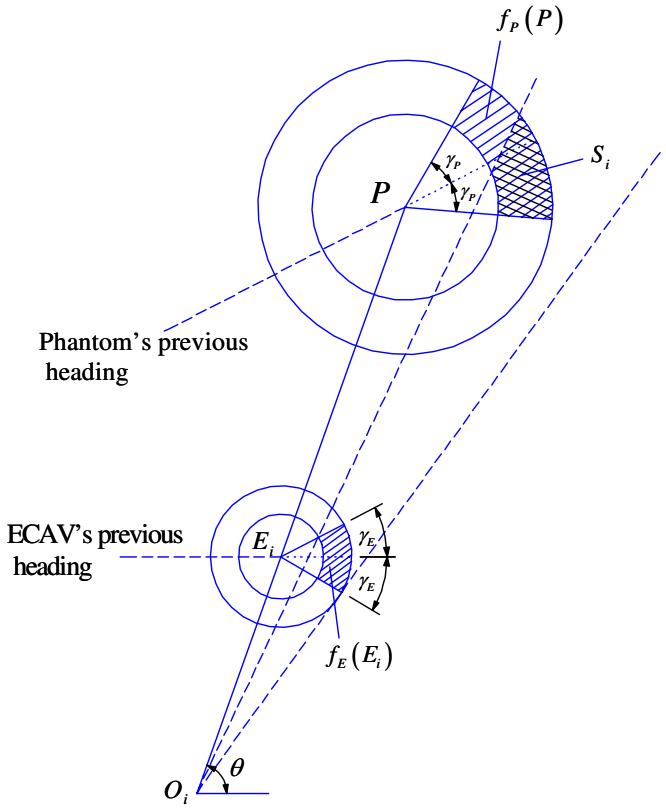

Fig. 16.: *Feasible velocity sectors of the phantom and the ECAV for bounded ground speeds and turn rates*

The bounds on the turn rates are $\pm\gamma_P$ and $\pm\gamma_E$ for the phantom and the ECAV respectively and with a unit incremental time step assumed, they become the angles defining the feasible velocity sectors $f_P(P)$ and $f_E(E)$. Thus the phantom can instantaneously change the course it has at time t by a maximum of $\pm\gamma_P$ and likewise the ECAV a maximum of

$\pm\gamma_E$. The set $S_i \subset f_P(P)$ the phantom can choose its next waypoint that ensures existence of solutions for the ECAV is the region cross hatched in Fig.16.

1. Simulation results for the case of bounds on ground speeds and turn rates

The bounds on the ground speeds of the ECAV and the phantom and the incremental time step of the algorithm are all unaltered from our earlier discussion of results. Turn rate bounds on the ECAVs and the phantom are introduced through minimum turn radii. A minimum turn radius of $6000m$ is assumed for the phantom while a $2000m$ minimum turn radius is assumed for each of the ECAVs. The minimum turn radius of $2000m$ translates into approximate turn rate bounds of $\pm 5 degrees/sec.$ Simulation results for the case of four ECAVs cooperatively deceiving four radars in a network by generating a coherent phantom track is shown in Fig.17. As seen from the simulation results all the ECAV trajectories appear realistic. Again the phantom locks onto a straight line path once all restrictions on the feasible velocity sector of the phantom relaxes and the phantom smoothly curves towards the final desired waypoint. The removal of all restrictions on the feasible velocity sector corresponds with all ECAVs settling into favorable positions on the perceived range vectors of each of their radars given by eq.(6.8). This results in the parallel flight of all ECAVs and the phantom as can be seen in Fig.17

With the introduction of the turn rate bounds, a feasible solution may not always be found for certain initial conditions. In fact when the ECAVs are placed too close to the initial waypoint of the phantom on their range vectors the algorithm sometimes came to a stop due to the set S falling empty. For certain initial and final waypoints the trajectory failed to connect the two within a reasonable time period. However instances of initial conditions where the algorithm failed to generate a phantom trajectory connecting the initial to the final waypoint were by far the exceptions rather than the rule.

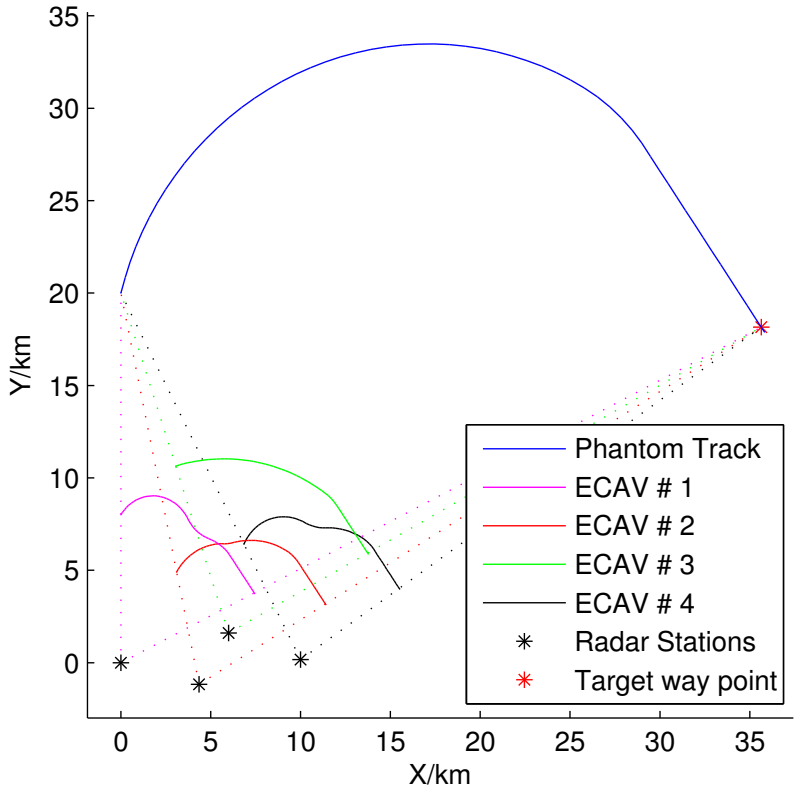

Fig. 17.: *Simulation results for a team of four ECAVs engaging four radars for the case of bounds on ground speeds and turn rates.*

CHAPTER VII

CONCLUSIONS AND FUTURE WORK

This study presents an algorithm that generates trajectories through a set of waypoints for a team of cooperating ECAVs to deceive a radar network by creating a phantom track. Trajectory planning of ECAVs to generate a phantom track is essentially a constrained optimization problem with two point boundary values with an unknown final time. The fact that each ECAV has to maintain collinearity with its radar and a phantom common to all ECAVs makes it a tightly coupled trajectory planning problem. Since solving a constrained optimization problem with two point boundary values is computationally intense at best, we have developed in this study an algorithm that generates sub optimal but feasible solutions. The Algorithm is initially formulated for the case of a single ECAV engaging a single radar, but the approach scales to n-ECAVs engaging n-radars. The fact that the algorithm implements finite dimensional searches makes the approach computationally attractive and amenable to real time computations. The approach followed is different to earlier work on this same problem in that we do not assume a pre-determined phantom track.

The algorithm generating trajectories through a finite set of waypoints can be considered as solving a sequence of constrained optimization problems, over discrete time periods. This simplifies each sub problem into a constrained optimization with only initial boundary values with known final time where only time invariant solutions are sought for. The solutions that result, though they maybe optimal in each discrete time interval, will not in general be optimal over the entire duration of time the algorithm is executed for. The radar deception scenario addressed is not meant to be merely of operational significance but is more importantly intended to illustrate salient features of the algorithmic approach to cooperative control of ECAVs in *Electronic Warfare*.

The algorithm can be implemented in a decentralized manner. Its implementation is

successful for the problem restricted to the plane and the next major step is to generalize it to three dimensions. The three dimensional case will have to consider the altitude of the phantom, the ECAVs and the radar in addition to their range and azimuth. The consideration of bounds on acceleration, collision avoidance and maintaining communication connectivity of the ECAVs is not addressed in this work and will be considered in future work.

REFERENCES

[1] P.Z. Peebles, *Radar Principles*, New York: John Wiley & Sons, Inc, 1998.

[2] G.W. Stimson, *Introduction to Airborne Radar*, 2nd edition, Raleigh, NC: SciTech Publishing, 1998.

[3] D.C. Schleher, *Electronic Warfare*, New York: Artech House, 1986.

[4] M. Pachter, P.R. Chandler, R.A. Larson, K.B. Purvis, "Concepts for Generating Coherent Radar Phantom Tracks Using Cooperative Vehicles", presented at the *AIAA Guidance, Navigation and Contr. Conf.*, Providence, RI, August 2004.

[5] K.B. Purvis, P.R. Chandler, and M. Pachter, "Feasible Flight Paths for Cooperative Generation of a Phantom Radar Track", presented at the *AIAA Guidance, Navigation and Contr. Conf.*, Providence, RI, August 2004.

[6] W.R. Beard and W.T. McLain, "Multiple UAV Cooperative Search under Collision Avoidance and Limited Range Communication Constraints", in *Proc. 42nd IEEE Conf. Decision and Control*, Maui, HI, December 2003, pp. 25-30.

APPENDIX A

MATLAB CODE FOR THE PHANTOM TRACK GENERATION

```
%initial conditions (R1P,R1P2,R1R2,R1R3,R1R4,R1O1,R2O2,R3O3,R4O4)
%and speed bounds of the ECAVs and the Phantom figure-1

%Entry (1): initial and final way points of the Phantom
%           initial way-point of the phantom
teta1_deg=90;        %theta angle of R1P in DEGREES
r1p= 20;             %\R1P\ in km.....this is the initial
%distance from the radar#1 to the phantom point final way-point
%of the phantom
tetaP2_deg=20;       %angle of R1P2 in DEGREES
r1P2=47 ;            %\R1P2\ in km....this is the distance from
%the radar#1 to final point of the phantom point

%Entry (2): radar locations
%           for radar point #1
%this radar sits at the origin of the polar coordinate system
%           for radar point # 2
r1r2=4;              %\R1R2\ in km....this is the distance from
%the radar#1 to radar#2
tetar1r2_deg=-3;     %theta angle of R1R2 in DEGREES
%           for radar point # 3
r1r3=7;              %\R1R3\ in km....this is the distance from
```

```
%the radar#1 to radar#3
tetar1r3_deg=8;      %theta angle of R1R3 in DEGREES
%           for radar point # 4
r1r4=8;              %\R1R4\ in km....this is the distance from
%the radar#1 to radar#4
tetar1r4_deg=1;      %theta angle of R1R4 in DEGREES

%Entry (3): ECAV locations
r1o1=3;             %\R1O1\ in km....this is the initial
%distance to the ECAV#1 from radar#1
r2o2=10;            %\R2O2\ in km....this is the initial
%distance to the ECAV#2 from radar#2
r3o3=5.2;           %\R3O3\ in km....this is the initial
%distance to the ECAV#3 from radar#3
r4o4=5.2;           %\R4O4\ in km....this is the initial
%distance to the ECAV#4 from radar#4

%Entry (4); speed bounds for the phantom and the ECAVs(in m/s)
%           speed bounds for the phantom
vmax=440;           %upper bound for speed
vmin=380;           %lower bound for speed
%           speed bounds for the ECAVs (in m/s)
vomax=115;          %upper bound for speed
vomin=85;           %lower bound for speed

%Entry (5): lower bounds on the minimum turn radius of the
```

```
%phantom and the ECAVs
%           lower bound on the turn radius of the Phantom
PH_turnradius=6000;   %turn radius in meters
%           lower bound on the turn radius of the ECAVs
ECAV_turnradius=2000; %turn radius in meters

%Entry (6): time step of the algorithm
f=1;                  %iteration factor (time step of the
%algorithm in seconds)

%Entry (7): range bound for the Phantom point
rpmax=200;            %upper bound for the range of the Phantom
%point from any of the radar points....in km

main;

%the main program

%range bounds for the ECAVs
romax=100;            %upper bound for the range of the ECAV
romin=0;             %lower bound for the range of the ECAV

vmax=vmax*10^-3*f;
vmin=vmin*10^-3*f;
vomax=vomax*10^-3*f;
vomin=vomin*10^-3*f;
```

```
PH_turnradius=PH_turnradius*10^-3;
ECAV_turnradius=ECAV_turnradius*10^-3;

resP=vmax;
oldheading=0;
OHresO1=0;
OHresO2=0;
OHresO3=0;
OHresO4=0;
ch1=1;
ch2=1;
ch3=1;
ch4=1;
del=norm(0.5*resP);
r=1;
LC=0;
localLC=0
linecount=200;

store_R1O1=[];
temp_R1O1=[];
store_resO1=[];
temp_resO1=[];
store_R1O2=[];
temp_R1O2=[];
```

```
store_resO2=[];
temp_resO2=[];
store_R1O3=[];
store_resO3=[];
store_R1O4=[];
store_resO4=[];
store_R1P=[];
temp_R1P=[];
store_resP=[];
temp_resP=[];

teta1=teta1_deg*pi/180;
tetaP2=tetaP2_deg*pi/180;
tetar1r2=tetar1r2_deg*pi/180;
tetar1r3=tetar1r3_deg*pi/180;
tetar1r4=tetar1r4_deg*pi/180;

turnrate=2*asin(vmax/(2*PH_turnradius));
ECAVturnrate=2*asin(vomin/(2*ECAV_turnradius));

tempturnrate=turnrate;
tempvmax=vmax;

[xR1P,yR1P]=pol2cart(teta1,r1p);
[xR1P2,yR1P2]=pol2cart(tetaP2,r1P2);
[xR1O1,yR1O1]=pol2cart(teta1,r1o1);
```

```
R1P=[xR1P,yR1P];
R1P2=[xR1P2,yR1P2];
R1O1=[xR1O1,yR1O1];

[xR1R2,yR1R2]=pol2cart(tetar1r2,r1r2);
R1R2=[xR1R2,yR1R2];
R2P2=R1P2-R1R2;
R2P=R1P-R1R2;
R2O2=R2P*r2o2/norm(R2P);
R1O2=R1R2+R2O2;

[xR1R3,yR1R3]=pol2cart(tetar1r3,r1r3);
R1R3=[xR1R3,yR1R3];
R3P2=R1P2-R1R3;
R3P=R1P-R1R3;
R3O3=R3P*r3o3/norm(R3P);
R1O3=R1R3+R3O3;

[xR1R4,yR1R4]=pol2cart(tetar1r4,r1r4);
R1R4=[xR1R4,yR1R4];
R4P2=R1P2-R1R4;
R4P=R1P-R1R4;
R4O4=R4P*r4o4/norm(R4P);
R1O4=R1R4+R4O4;
if (romin > norm(r1o1)) | (romin > norm(r2o2)) | (romin >
```

```
    norm(r3o3)) | (romin > norm(r4o4)) | (romax < norm(r1o1)) |
    (romax < norm(r2o2)) |(romax < norm(r3o3)) | (romax <
    norm(r4o4))
    'invalid initial conditions for the ECAVs'
elseif norm(R1P)>=rpmax | norm(R2P)>=rpmax | norm(R3P)>=rpmax |
    norm(R4P)>=rpmax
    'invalid initial conditions for the Phantom point'
elseif norm(r1P2)>=rpmax | norm(R2P2)>=rpmax | norm(R3P2)>=rpmax
    | norm(R4P2)>=rpmax
    'target point is beyond the range bound of the Phantom point'
else
    %plotting dotted lines that join each radar point to P
    hold on;
    plot([0,R1P(1,1)],[0,R1P(1,2)],':m');
    plot([R1R2(1,1),R1P(1,1)],[R1R2(1,2),R1P(1,2)],':r');
    plot([R1R3(1,1),R1P(1,1)],[R1R3(1,2),R1P(1,2)],':g');
    plot([R1R4(1,1),R1P(1,1)],[R1R4(1,2),R1P(1,2)],':k');
    plot([0,R1P2(1,1)],[0,R1P2(1,2)],':m');
    plot([R1R2(1,1),R1P2(1,1)],[R1R2(1,2),R1P2(1,2)],':r');
    plot([R1R3(1,1),R1P2(1,1)],[R1R3(1,2),R1P2(1,2)],':g');
    plot([R1R4(1,1),R1P2(1,1)],[R1R4(1,2),R1P2(1,2)],':k');
    %plotting the three radar points and the final desired
    %location of point P
    h7=plot(R1P2(1,1),R1P2(1,2),'*r');
    h6=plot(0,0,'*k');
    plot(R1R2(1,1),R1R2(1,2),'*k');
```

```
plot(R1R3(1,1),R1R3(1,2),'*k');
plot(R1R4(1,1),R1R4(1,2),'*k');

PP2=R1P2-R1P;
initPP2=norm(PP2);

while (norm(PP2)>del) & (norm(PP2)<norm(initPP2)*10)  & LC<
    linecount & r==1 & norm(R1P)<rpmax & norm(R2P)<rpmax  &
    norm(R3P)<rpmax & norm(R4P)<rpmax
    LC=LC+1
    del=norm(0.5*resP);

    RP=[[R1P(1,1);R1P(1,2)],[R2P(1,1);R2P(1,2)],[R3P(1,1);
        R3P(1,2)],[R4P(1,1);R4P(1,2)],];
    RO=[[R1O1(1,1);R1O1(1,2)],[R2O2(1,1);R2O2(1,2)],
        [R3O3(1,1);R3O3(1,2)],[R4O4(1,1);R4O4(1,2)],];
    v=[vmax,vmin,vomax,vomin];

    %get gema angles at P that guarantees solutions for each
    %ECAV
    [gema]=get_gema(v,RO,RP);

    %getting RPt
    [RPt]=get_RPt(RP);
    R1Pt=[RPt(1,1),RPt(2,1)];
    R2Pt=[RPt(1,2),RPt(2,2)];
```

```
R3Pt=[RPt(1,3),RPt(2,3)];
R4Pt=[RPt(1,4),RPt(2,4)];

%getting the points that define the boundary of the
%feasible velocity sector for P
[Mu]=boundry_points(RP,RPt,gema,v);

%checking if target is within the allowable vel. sector
[V,in]=calresP(RP,RPt,PP2,gema,v);

if in==1
    resP=V;
else
    %getting the closest point to P2
    [resP]=closestpoint(Mu,PP2);
end
tempresP=resP;
%checking if any of the ECAVs are within the range
[within]=check_ECAV_within_range(RP,RO,v);

if within==1 & LC~=1
    if ECAVturnrate < turnrate
        [resP]=check_tetaddot(resP,oldheading,
        ECAVturnrate-10^-4,RP,gema);
    else
        [resP]=check_tetaddot(resP,oldheading,turnrate,
```

```
        RP,gema);
    end
elseif LC~=1
    [resP]=check_tetaddot(resP,oldheading,turnrate,RP,
    gema);
end

ch1=1;
while (ch1==1 | ch2==1 | ch3==1 | ch4==1) & localLC<3
    localLC=localLC+1;
    'ecav#1'
    [resO1,N1,resP,ch1,gema,v]=calresO(R1O1,R1P,R1Pt,
    resP,v,romin,romax,OHresO1,ECAVturnrate,oldheading,
    turnrate,PP2,RP,RPt,RO,gema);

    'ecav#2'
    [resO2,N2,resP,ch2,gema,v]=calresO(R2O2,R2P,R2Pt,
    resP,v,romin,romax,OHresO2,ECAVturnrate,oldheading,
    turnrate,PP2,RP,RPt,RO,gema);

    'ecav#3'
    [resO3,N3,resP,ch3,gema,v]=calresO(R3O3,R3P,R3Pt,
    resP,v,romin,romax,OHresO3,ECAVturnrate,oldheading,
    turnrate,PP2,RP,RPt,RO,gema);

    'ecav#4'
```

```
    [resO4,N4,resP,ch4,gema,v]=calresO(R4O4,R4P,R4Pt,
    resP,v,romin,romax,OHresO4,ECAVturnrate,oldheading,
    turnrate,PP2,RP,RPt,RO,gema);
end
localLC=0;

if norm(N1)==0 | norm(N2)==0 | norm(N3)==0 | norm(N4)==0
    r=0;
end

if norm(N1)~=0 & norm(N2)~=0 & norm(N3)~=0 & norm(N4)~=0
    store_R1O1=[store_R1O1,[R1O1(1,1);R1O1(1,2)]];
    temp_R1O1=[temp_R1O1;[R1O1(1,1),R1O1(1,2)]];
    store_resO1=[store_resO1,[resO1(1,1);resO1(1,2)]];
    temp_resO1=[temp_resO1;[resO1(1,1),resO1(1,2)]];
    OHresO1=resO1;
    store_R1O2=[store_R1O2,[R1O2(1,1);R1O2(1,2)]];
    temp_R1O2=[temp_R1O2;[R1O2(1,1),R1O2(1,2)]];
    store_resO2=[store_resO2,[resO2(1,1);resO2(1,2)]];
    temp_resO2=[temp_resO2;[resO2(1,1),resO2(1,2)]];
    OHresO2=resO2;
    store_R1O3=[store_R1O3,[R1O3(1,1);R1O3(1,2)]];
    store_resO3=[store_resO3,[resO3(1,1);resO3(1,2)]];
    OHresO3=resO3;
    store_R1O4=[store_R1O4,[R1O4(1,1);R1O4(1,2)]];
    store_resO4=[store_resO4,[resO4(1,1);resO4(1,2)]];
```

```
            OHresO4=resO4;

            oldheading=resP;
            turnrate=tempturnrate;

            store_R1P=[store_R1P,[R1P(1,1);R1P(1,2)]];
            temp_R1P=[temp_R1P;[R1P(1,1),R1P(1,2)]];
            store_resP=[store_resP,[resP(1,1);resP(1,2)]];
            temp_resP=[temp_resP;[resP(1,1),resP(1,2)]];

            R1O1=R1O1+resO1;
            R2O2=R2O2+resO2;
            R3O3=R3O3+resO3;
            R4O4=R4O4+resO4;
            R1O2=R1R2+R2O2;
            R1O3=R1R3+R3O3;
            R1O4=R1R4+R4O4;
            R1P=R1P+resP;
            R2P=R1P-R1R2;
            R3P=R1P-R1R3;
            R4P=R1P-R1R4;
            PP2=R1P2-R1P;
        end
    end
end
```

```
if norm(PP2)<del;
    'P is at P2'
elseif r==0
    'no solution exists for one or more of the ECAVs'
elseif LC>=linecount;
    'not converging fast enough'
elseif norm(R2P)>=rpmax | norm(R1P)>=rpmax | norm(R3P)>=rpmax |
    norm(R4P)>=rpmax
    'the Phantom point has gone beyond its range bound'
else
    'P2 is not reachable'
end
animation;


%get gema angles at P that guarantees solutions for each ECAV
function [gema]=get_gema(v,RO,RP)

vmax=v(1,1);
vomax=v(1,3);
R1O1=[RO(1,1),RO(2,1)];
R2O2=[RO(1,2),RO(2,2)];
R3O3=[RO(1,3),RO(2,3)];
R4O4=[RO(1,4),RO(2,4)];

R1P=[RP(1,1),RP(2,1)];
```

```
R2P=[RP(1,2),RP(2,2)];
R3P=[RP(1,3),RP(2,3)];
R4P=[RP(1,4),RP(2,4)];

gema1=asin(vomax/norm(R1O1)) + asin( norm(R1P)*sin( asin
(vomax/norm(R1O1))/(vmax) ) )-10^-4;
gema2=asin(vomax/norm(R2O2)) + asin( norm(R2P)*sin( asin
(vomax/norm(R2O2))/(vmax) ) )-10^-4;
gema3=asin(vomax/norm(R3O3)) + asin( norm(R3P)*sin( asin
(vomax/norm(R3O3))/(vmax) ) )-10^-4;
gema4=asin(vomax/norm(R4O4)) + asin( norm(R4P)*sin( asin
(vomax/norm(R4O4))/(vmax) ) )-10^-4;

if norm(imag(gema1))~=0 | norm([RP(1,1),RP(2,1)])*vomax/vmax >
    norm([RO(1,1),RO(2,1)])
    gema1=pi;
end
if norm(imag(gema2))~=0 | norm([RP(1,2),RP(2,2)])*vomax/vmax >
    norm([RO(1,2),RO(2,2)])
    gema2=pi;
end
if norm(imag(gema3))~=0 | norm([RP(1,3),RP(2,3)])*vomax/vmax >
    norm([RO(1,3),RO(2,3)])
    gema3=pi;
end
if norm(imag(gema4))~=0 | norm([RP(1,4),RP(2,4)])*vomax/vmax >
```

```
    norm([RO(1,4),RO(2,4)])
    gema4=pi;
end

gema=[gema1,gema2,gema3,gema4];

%getting RPt
function [RPt]=get_RPt(RP);

R1P=[RP(1,1),RP(2,1)];
R2P=[RP(1,2),RP(2,2)];
R3P=[RP(1,3),RP(2,3)];
R4P=[RP(1,4),RP(2,4)];

[teta1,r1p]=cart2pol(R1P(1,1),R1P(1,2));
[xR1Pt,yR1Pt]=pol2cart((teta1+90*pi/180),1);
R1Pt=[xR1Pt,yR1Pt];

[teta2,r2p]=cart2pol(R2P(1,1),R2P(1,2));
[xR2Pt,yR2Pt]=pol2cart((teta2+90*pi/180),1);
R2Pt=[xR2Pt,yR2Pt];    %tangent to RP

[teta3,r3p]=cart2pol(R3P(1,1),R3P(1,2));
[xR3Pt,yR3Pt]=pol2cart((teta3+90*pi/180),1);
R3Pt=[xR3Pt,yR3Pt];    %tangent to RP
```

```
[teta4,r4p]=cart2pol(R4P(1,1),R4P(1,2));
[xR4Pt,yR4Pt]=pol2cart((teta4+90*pi/180),1);
R4Pt=[xR4Pt,yR4Pt];    %tangent to RP

RPt=[[R1Pt(1,1);R1Pt(1,2)],[R2Pt(1,1);R2Pt(1,2)],[R3Pt(1,1);
    R3Pt(1,2)],[R4Pt(1,1);R4Pt(1,2)],];

%getting all the points that define the bounderies of the
%velocity sector from the necessary condition
function [M]=boundry_points(RP,RPt,gema,v)

vmax=v(1,1);
m1=[];
m2=[];
[m,n]=size(RP);
for i=1:n
    RP1=[RP(1,i),RP(2,i)];
    RP1t=[RPt(1,i),RPt(2,i)];
    gema1=gema(1,i);
    if gema1~=pi
        [Mupper,Mlower,gema_1]=points_circle(RP1,RP1t,gema1,
        vmax);
        m1=[m1,Mupper];
    end
 end
```

```
%updating M with points that lie in the common velocity sector
if norm(m1)~=0
    [M]=updatingM(m1,gema,RP);
else
    M=m1;
end
```

```
%getting all the points that lie on the cricumference of the
%speed bound into M all points will be with respect to point P
function [Mupper,Mlower,gema_point]=points_circle(RP,RPt,gema,V)

point1=RP*V*cos(gema)/norm(RP)-RPt*V*sin(gema)/norm(RPt);
beta=atan( V*sin(gema)/(norm(RP)+V*cos(gema)) );
delta=gema-beta;
delt=delta*180/pi;
gema_point2=gema+pi-2*delta;
point2=RP*V*cos(gema_point2)/norm(RP)-RPt*V*sin(gema_point2)
/norm(RPt);
point4=RP*V*cos(gema)/norm(RP)+RPt*V*sin(gema)/norm(RPt);
point3=RP*V*cos(gema_point2)/norm(RP)+RPt*V*sin(gema_point2)
/norm(RPt);
Mupper=[point1(1,1) point4(1,1);point1(1,2) point4(1,2)];
```

```
Mlower=[point2(1,1) point3(1,1);point2(1,2) point3(1,2)];

gema_point=gema_point2;

%updating M with points that lie within the common velocity
%region
function [M]=updatingM(M,gema,RP)
tol=10^-6;
[r,s]=size(gema);
[x,y]=size(M);
m=[];

for j=1:s
    m=[];

    gema1=gema(1,j);
    RP1=[RP(1,j),RP(2,j)];
    for i=1:y
        if (acos(dot(RP1,[M(1,i),M(2,i)])/(norm(RP1)*norm(
            [M(1,i),M(2,i)]))) <= gema1 +tol)
                m=[m,[M(1,i);M(2,i)]];
        end
    end
    M=m;
    [x,y]=size(M);
end
```

```
%checking if the target point lies within the feasible velocity
%sector of P
function [V,in]=calresP(RP,RPt,PP2,gema,v)

vmax=v(1,1);
in=1;
V=[0,0];
[m,n]=size(RP);
for i=1:n
    RP1=[RP(1,i),RP(2,i)];
    RP1t=[RPt(1,i),RPt(2,i)];
    alpha=acos(dot(PP2,RP1)/(norm(RP1)*norm(PP2)));

    if norm(alpha)<norm(gema(1,i))
        if (dot(RP1t,PP2)>=0) %& (dot(RP,PP2)>=0)
            V=vmax*sin(alpha)*RP1t/norm(RP1t) + vmax*cos(
            lpha)*RP1/norm(RP1);
        else
            V=-vmax*sin(alpha)*RP1t/norm(RP1t) + vmax*cos(
            alpha)*RP1/norm(RP1);
        end
    else
        in=0;
    end
```

```
end

%getting the closest point to the target from the matrix M
function [closerintpoint]=closestpoint(M,PP2)

closerintpoint=[M(1,1),M(2,1)];
[m,n]=size(M);
for i=1:n
    for j=i:n
        if i~=j
            if norm([M(1,i)-PP2(1,1),M(2,i)-PP2(1,2)]) < norm(
                [M(1,j)-PP2(1,1),M(2,j)-PP2(1,2)])
                if norm([M(1,i)-PP2(1,1),M(2,i)-PP2(1,2)]) <
                    norm([closerintpoint(1,1)-PP2(1,1),
                        closerintpoint(1,2)-PP2(1,2)])
                    closerintpoint=[M(1,i),M(2,i)];
                end
            else
                if norm([M(1,j)-PP2(1,1),M(2,j)-PP2(1,2)]) <
                    norm([closerintpoint(1,1)-PP2(1,1),
                        closerintpoint(1,2)-PP2(1,2)])
                    closerintpoint=[M(1,j),M(2,j)];
                end
            end
        end
    end
```

```
end

%checking if any of the ECAVs are within that range that removes
%all restrictions at the velocity sector at P
function [anyECAVwithin]=check_ECAV_within_range(RP,RO,v)

[m,n]=size(RP);
anyECAVwithin=0;

for i=1:n
    if norm([RP(1,i),RP(2,i)])*v(1,3)/v(1,1) > norm([RO(1,i),
            RO(2,i)]) & norm([RO(1,i),RO(2,i)])> norm([RP(1,i),
            RP(2,i)])*v(1,4)/v(1,1)
        anyECAVwithin=1;
    end
end

%calculatinf the resultant velocity for the Phanrom point keeping
%it within the bounds on the turn rate
function [resP]=check_tetaddot(resP,oldheading,turnrate,RP,gema)

[oldheading_teta,oldheading_r]=cart2pol(oldheading(1,1),
oldheading(1,2));
[oldheading_t_x,oldheading_t_y]=pol2cart((oldheading_teta+pi/2),
oldheading_r);
oldheading_t=[oldheading_t_x,oldheading_t_y];
turnangle=acos(dot(resP,oldheading)/(norm(resP)*norm(oldheading
```

```
)));

if turnangle>turnrate
    sgn=sign(dot(resP,oldheading_t));
    resP=norm(resP)*cos(turnrate)*oldheading/norm(oldheading) +
    norm(resP)*sin(turnrate)*oldheading_t*sgn/norm(oldheading_t);
    %cheking if resP is still within the feasible sector
    [check]=check_resP(resP,RP,gema);
    if check==0
        resP=norm(resP)*cos(turnrate)*oldheading/norm(oldheading)
        - norm(resP)*sin(turnrate)*oldheading_t*sgn/norm(
        oldheading_t);
    end
else
    resP=resP;
end



%calculating the resultant velocity for each UAV
function [resO,N,resP,change,gema,v]=calresO(RO,RP,RPt,resP,v,
romin,romax,OHresO,ECAVturnrate,oldheading,turnrate,PP2,RP_tot,
RPt_tot,RO_tot,gema)

vmax=v(1,1);
vmin=v(1,2);
```

```
vomax=v(1,3);
vomin=v(1,4);
change=0;
within=0;
check=1;
if norm(RP)*vomax/vmax > norm(RO) & norm(RO)> norm(RP)*vomin/vmax
    within=1;
    r1pdot=(dot(resP,RP)/norm(RP))*RP/norm(RP);
    r1o1dot=(norm(RO)/norm(RP))*r1pdot;
    [teta1,r1p]=cart2pol(RP(1,1),RP(1,2));
    [xRPt,yRPt]=pol2cart((teta1+90*pi/180),1);
    RPt=[xRPt,yRPt];
    r1ptetadot=(dot(resP,RPt)/norm(RPt))*RPt/norm(RPt);
    r1o1tetadot=r1ptetadot*norm(RO)/norm(RP);
    resO=r1o1dot+r1o1tetadot;
    if OHresO~=0 %& norm(resO+RO)<romax & norm(resO+RO)>romin
        turnangle=acos(dot(resO,OHresO)/(norm(resO)*norm(OHresO)
        ));
        if turnangle<=ECAVturnrate
            check=0;
            N=[1;1];
        end
    end
end
if check==1
    %getting all the intersection points of the line RP with
```

```
%the two velocity circles at O
[N1]=intpoints_resO(RO,RP,resP,v);

%updating N with upper intersection points only
[N2]=upper_intpoints_resO(RO,RP,v,N1);

%if the ECAV is within the range or if it has only two points
then N1
%is retained...else N2 is chosen. N is updated with the point
that lies on RP that gives
%the required ratio IF it is in the velocity sector of the
ECAV
[N3]=updating_N_with_possible_better_intermediate_points(N1,
N2,resP,RP,RO,v,within,OHresO);

%update this N with turnrate bounds with respect to the
oldheading if
%they happen to lie between the two extreame points of N
[N4]=updating_N_ECAV_turnrate_bounds(N3,RO,OHresO,
ECAVturnrate,v);

%updating N with points that don't violate the turnrate
constraint
[N]=check_ECAV_turnrate(N4,RO,OHresO,ECAVturnrate);

%getting the point (of N) that is closest to the ratio of
```

```
%0.5*(vomax+vomin)/vmax
if norm(N)~=0
    [resO]=better_intpoint_resO(RO,RP,resP,v,N);
else

    %choose the point that is closest to the turnrate bounds
    [resO]=point_closest_to_turnrate(RO,OHresO,N4);

    %if N is empty, try the following..first try decreasing
    %the iteration time..then try getting resP corresponding
    %to an allowable resO and check if this resP is within
    %its allowable region and if so also if its within its
    %turnrate bounds.....if that doesn;t work either then
    %try increasing iteration time...

    %getting an allowable resO and then check if the
    %corresponding resP is acceptable
    [N,resO,resP]=resP_for_allowable_resO(N,v,resP,resO,RO,
    RP,OHresO,ECAVturnrate,RP_tot,gema,turnrate,oldheading);
    if norm(N)==0
        'getting a resP corresponding to an allowable resO
        didnt work'
        %try decreasing the iteration time
        [N,resO,resP,v,gema,ECAVturnrate,turnrate]=
        decrease_iteration_time(N,v,resP,resO,RO,RP,OHresO,
        ECAVturnrate,PP2,RP_tot,RPt_tot,RO_tot,gema,turnrate,
```

```
            oldheading,within);
            if norm(N)==0
                'decreasing the interation time didnt help'
                %try increasing the iteration time
                [N,resO,resP,v,gema,ECAVturnrate,turnrate]=
                increase_iteration_time(N,v,resP,resO,RO,RP,
                OHresO,ECAVturnrate,PP2,RP_tot,RPt_tot,RO_tot,
                gema,turnrate,oldheading,within);
                if norm(N)==0
                    'increasing the iteration time didnt help
                    either'
                else
                    change=1;
                end
            else
                change=1;
            end
        else
            change=1;
        end
    end
end
```

```
%getting all the intersection points of the line joining R to P
%with the two velocity circles
function [N1]=intpoints_resO(RO,RP,resP,v)

vmax=v(1,1);
vmin=v(1,2);
vomax=v(1,3);
vomin=v(1,4);

RP_resP=RP+resP;
%y=m*x+c
m=RP_resP(1,2)/RP_resP(1,1);
c=0;
a=RO(1,1);
b=RO(1,2);

x=[ 1/2/(1+m^2)*(-2*m*c+2*b*m+2*a+2*(-2*m*c*a+2*b*m*a-b^2-c^2+
2*b*c+vomax^2-m^2*a^2+m^2*vomax^2)^(1/2)),1/2/(1+m^2)*(-2*m*c+
2*b*m+2*a-2*(-2*m*c*a+2*b*m*a-b^2-c^2+2*b*c+vomax^2-m^2*a^2+
m^2*vomax^2)^(1/2))];
y=[ 1/2*m/(1+m^2)*(-2*m*c+2*b*m+2*a+2*(-2*m*c*a+2*b*m*a-b^2-c^2+
2*b*c+vomax^2-m^2*a^2+m^2*vomax^2)^(1/2))+c,1/2*m/(1+m^2)*(
-2*m*c+2*b*m+2*a-2*(-2*m*c*a+2*b*m*a-b^2-c^2+2*b*c+vomax^2-
m^2*a^2+m^2*vomax^2)^(1/2))+c];

N1=[];
```

```
if norm(imag(x))==0
    N1=[N1,[x(1,1);y(1,1)],[x(1,2);y(1,2)]];
end

x=[1/2/(1+m^2)*(-2*m*c+2*b*m+2*a+2*(-2*m*c*a+2*b*m*a-b^2-c^2+
2*b*c+vomin^2-m^2*a^2+m^2*vomin^2)^(1/2)), 1/2/(1+m^2)*(-2*m*c+
2*b*m+2*a-2*(-2*m*c*a+2*b*m*a-b^2-c^2+2*b*c+vomin^2-m^2*a^2+
m^2*vomin^2)^(1/2))];
y=[ 1/2*m/(1+m^2)*(-2*m*c+2*b*m+2*a+2*(-2*m*c*a+2*b*m*a-b^2-c^2+
2*b*c+vomin^2-m^2*a^2+m^2*vomin^2)^(1/2))+c, 1/2*m/(1+m^2)*(
-2*m*c+2*b*m+2*a-2*(-2*m*c*a+2*b*m*a-b^2-c^2+2*b*c+vomin^2-
m^2*a^2+m^2*vomin^2)^(1/2))+c];
if norm(imag(x))==0
    N1=[N1,[x(1,1);y(1,1)],[x(1,2);y(1,2)]];
end

%updating N with upper intersection points only
function [N2]=upper_intpoints_resO(RO,RP,v,N1)

vomax=v(1,3);
N2=[];
    [m,n]=size(N1);
    [teta1,r1p]=cart2pol(RP(1,1),RP(1,2));
    [xRPt,yRPt]=pol2cart((teta1+90*pi/180),1);
    RPt=[xRPt,yRPt];
```

```
    delta=asin(vomax/norm(RO));
    sgn_del=sign(dot(RPt,[N1(1,1),N1(2,1)])/(norm(RPt)*norm(
    [N1(1,1),N1(2,1)])));

    [xRP_del,yRP_del]=pol2cart((teta1+sgn_del*delta),1);
    RP_minusdelta=[xRP_del,yRP_del] ;
    for i=1:n
        sgn=dot([N1(1,i),N1(2,i)]-RO,RP_minusdelta);
        if sgn>=0
            N2=[N2,[N1(1,i);N1(2,i)]];
        end
    end

%if the required ratio is given by a point that lies between two
%points of N then N is updated with that point as well
function [N3]=
updating_N_with_possible_better_intermediate_points(
N1,N2,resP,RP,RO,v,within,OHresO)

vmax=v(1,1);
vomax=v(1,3);
vomin=v(1,4);

RP_resP=RP+resP;

ROcr=norm(RP_resP)*0.5*(vomax+vomin)/vmax;
```

```
R_ROcr=ROcr*RP_resP/norm(RP_resP);
O_ROcr=R_ROcr-RO;

[m,n]=size(N1);
if (n==2 | within==1) & OHresO~=0
    N=N1;
else
    N=N2;
end

N3=N;
if ( vomin < norm(O_ROcr) ) & ( norm(O_ROcr) < vomax )
    N3=[N,[R_ROcr(1,1);R_ROcr(1,2)]];
end



%if N1 has 4 points pick N2 ..if N1 has only two points retain N1
%update this N with turnrate bounds with respect to the
%oldheading if they happen to lie between the two points of N
function [N3]=updating_N_ECAV_turnrate_bounds(N,RO,OHresO,
ECAVturnrate,v)

ECAVturnrate=ECAVturnrate-10^-4;
vomax=v(1,3);
vomin=v(1,4);
```

```
if OHresO~=0
    %y=mx is the line through the radar point and points of N
    %y=m1x+c1 and y=m2x+c2 will be the lines through O that
    %define the turnrate bounds

    m=N(2,1)/N(1,1);

    [teta1,r1ohresO]=cart2pol(OHresO(1,1),OHresO(1,2));
    [xOHresO,yOHresO]=pol2cart((teta1+90*pi/180),1);
    OHresOt=[xOHresO,yOHresO];

    b1=norm(OHresO)*cos(ECAVturnrate)*OHresO/norm(OHresO)+norm(
    OHresO)*sin(ECAVturnrate)*OHresOt/norm(OHresOt);
    b2=norm(OHresO)*cos(ECAVturnrate)*OHresO/norm(OHresO)-norm(
    OHresO)*sin(ECAVturnrate)*OHresOt/norm(OHresOt);

    m1=b1(1,2)/b1(1,1);
    c1=RO(1,2)-m1*RO(1,1);
    m2=b2(1,2)/b2(1,1);
    c2=RO(1,2)-m2*RO(1,1);

    %getting intersection points (x1,y1) and (x2,y2) of y=mx with
    %y=m1x+c1 and y=m2x+c2
    x1=c1/(m-m1);
    y1=m*x1;
```

```
    x2=c2/(m-m2);
    y2=m*x2;

    O_P1=[x1,y1]-RO;
    O_P2=[x2,y2]-RO;

    N3=N;
    if vomin < norm(O_P1) & norm(O_P1) < vomax
        N3=[N3,[x1;y1]];
    end
    if vomin < norm(O_P2) & norm(O_P2) < vomax
        N3=[N3,[x2;y2]];
    end
else
    N3=N;
end

%calculatinf the resultant velocity for the Phanrom point keeping
%it within the bounds on the turn rate
function [N]=check_ECAV_turnrate(N,RO,oldheading,turnrate)
turnrate=turnrate+10^-4;
if oldheading~=0
    [m,n]=size(N);
    updatedN=[];
    for i=1:n
        tempresO=[N(1,i),N(2,i)]-RO;
```

```
        turnangle=acos(dot(tempresO,oldheading)/(norm(tempresO)*
        norm(oldheading)));
        if turnangle<turnrate
            updatedN=[updatedN,[N(1,i);N(2,i)]];
        end
    end
    N=updatedN;
end
```

```
%getting the point (of N) that is closest to the ratio of
%v1max/vo1max
function [resO]=better_intpoint_resO(RO,RP,resP,v,N)

RP_resP=RP+resP;
vmax=v(1,1);
vomax=v(1,3);
vomin=v(1,4);
ROcr=norm(RP_resP)*0.5*(vomax+vomin)/vmax;
```

```
        New_N=N;
        better_point=[New_N(1,1),New_N(2,1)] ;
        [m,n]=size(New_N);
        for i=1:n
            if norm(norm([N(1,i),N(2,i)])-ROcr) < norm(norm(
                better_point)-ROcr)
                better_point=[New_N(1,i),New_N(2,i)];
            end
        end
        resO=better_point-RO;

%choose the point that is closest to the turnrate bounds
function [resO]=point_closest_to_turnrate(RO,OHresO,N)

New_N=N;
better_point=[New_N(1,1),New_N(2,1)]-RO ;

[m,n]=size(New_N);
for i=1:n
    %gettting the vector from O to points of N
    O_N=[N(1,i),N(2,i)]-RO;
    %getting the angle between that and the OHresO
    angle=acos( dot(O_N,OHresO)/(norm(O_N)*norm(OHresO)) );
    angle_last=acos( dot(better_point,OHresO)/(norm(better_point)
    *norm(OHresO)) );
    if angle<angle_last
```

```
        better_point=[New_N(1,i),New_N(2,i)];
    end
end
resO=better_point-RO;


%try getting an allowable resO and then check if the
%corresponding resP is acceptable
function [N,resO,resP]=resP_for_allowable_resO(N,v,resP,resO,RO,
RP,OHresO,ECAVturnrate,RP_tot,gema,turnrate,oldheading)
vmax=v(1,1);
vomax=v(1,3);
vomin=v(1,4);
tempresO=resO;

%get points A&B that are the intersection points of ECAV
%turnrates bounds with the smaller velocity circle of the ECAV
[teta1,r1ohresO]=cart2pol(OHresO(1,1),OHresO(1,2));
[xOHresO,yOHresO]=pol2cart((teta1+90*pi/180),1);
OHresOt=[xOHresO,yOHresO];

A=vomin*cos(ECAVturnrate)*OHresO/norm(OHresO)+vomin*sin(
ECAVturnrate)*OHresOt/norm(OHresOt);
B=vomin*cos(ECAVturnrate)*OHresO/norm(OHresO)-vomin*sin(
ECAVturnrate)*OHresOt/norm(OHresOt);
```

```
%pick the point that is closer to ROcr
ROcr=norm(RP)*0.5*(vomax+vomin)/vmax;
if norm(norm(RO+A)-ROcr) < norm(norm(RO+B)-ROcr)
    resO=A;
    resO1=B;
else
    resO=B;
    resO1=A;
end
%y=mx
RO_resO=RO+resO;
m=RO_resO(1,2)/RO_resO(1,1);
c=0;

%getting intersection points of y=mx with the larger velocity
%circle at P
a=RP(1,1);
b=RP(1,2);
x=[ 1/2/(1+m^2)*(-2*m*c+2*b*m+2*a+2*(-2*m*c*a+2*b*m*a-b^2-c^2+
2*b*c+vmax^2-m^2*a^2+m^2*vmax^2)^(1/2)),1/2/(1+m^2)*(-2*m*c+
2*b*m+2*a-2*(-2*m*c*a+2*b*m*a-b^2-c^2+2*b*c+vmax^2-m^2*a^2+
m^2*vmax^2)^(1/2))];
y=[ 1/2*m/(1+m^2)*(-2*m*c+2*b*m+2*a+2*(-2*m*c*a+2*b*m*a-b^2-c^2+
2*b*c+vmax^2-m^2*a^2+m^2*vmax^2)^(1/2))+c,1/2*m/(1+m^2)*(-2*m*c+
2*b*m+2*a-2*(-2*m*c*a+2*b*m*a-b^2-c^2+2*b*c+vmax^2-m^2*a^2+
m^2*vmax^2)^(1/2))+c];
```

```
N1=[];
if norm(imag(x))==0
    N1=[N1,[x(1,1);y(1,1)],[x(1,2);y(1,2)]];
else
    'no solution exists for resP'
end

%get points of N1 that lies within the allowable velocity sector
%at P
 N2=[];
if norm(N1)~=0
    for i=1:2
        [check]=check_resP([N1(1,i),N1(2,i)],RP_tot,gema);
        if check==1
            N2=[N2,[N1(1,i);N1(2,i)]];
        end
    end
end

%get points of N2 that does not violate the turnrate constraints
%of the phantom point
N3=[];
if norm(N2)~=0
    [m,n]=size(N2);
    for i=1:n
```

```
        P_N=[N2(1,i),N2(2,i)]-RP;
        turnangle=acos(dot(P_N,oldheading)/(norm(P_N)*norm(
        oldheading)));
        if turnangle<=turnrate
            N3=[N3,[N2(1,i);N2(2,i)]];
        end
    end
end

if norm(N3)==0
    %y=mx
    resO=resO1;
    RO_resO=RO+resO;
    m=RO_resO(1,2)/RO_resO(1,1);
    c=0;

    %getting intersection points of y=mx with the larger velocity
    %circle at P
    a=RP(1,1);
    b=RP(1,2);
    x=[ 1/2/(1+m^2)*(-2*m*c+2*b*m+2*a+2*(-2*m*c*a+2*b*m*a-b^2-
    c^2+2*b*c+vmax^2-m^2*a^2+m^2*vmax^2)^(1/2)),1/2/(1+m^2)*(
    -2*m*c+2*b*m+2*a-2*(-2*m*c*a+2*b*m*a-b^2-c^2+2*b*c+vmax^2-
    m^2*a^2+m^2*vmax^2)^(1/2))];
    y=[ 1/2*m/(1+m^2)*(-2*m*c+2*b*m+2*a+2*(-2*m*c*a+2*b*m*a-b^2-
    c^2+2*b*c+vmax^2-m^2*a^2+m^2*vmax^2)^(1/2))+c,1/2*m/(1+m^2)*
```

```
(-2*m*c+2*b*m+2*a-2*(-2*m*c*a+2*b*m*a-b^2-c^2+2*b*c+vmax^2-
m^2*a^2+m^2*vmax^2)^(1/2))+c];

N1=[];
if norm(imag(x))==0
    N1=[N1,[x(1,1);y(1,1)],[x(1,2);y(1,2)]];
else
    'no solution exists for resP...not even with the second
    choice for resO'
end

%get points of N1 that lies within the allowable velocity
%sector at P
if norm(N1)~=0
    N2=[];
    for i=1:2
        [check]=check_resP([N1(1,i),N1(2,i)],RP_tot,gema);
        if check==1
            N2=[N2,[N1(1,i);N1(2,i)]];
        end
    end
end

%get points of N2 that does not violate the turnrate
%constraints of the phantom point
N3=[];
```

```
    if norm(N2)~=0
        [m,n]=size(N2);
        for i=1:n
            P_N=[N2(1,i),N2(2,i)]-RP;
            turnangle=acos(dot(P_N,oldheading)/(norm(P_N)*norm(
            oldheading)));
            if turnangle<=turnrate
                N3=[N3,[N2(1,i);N2(2,i)]];
            end
        end
    end
end

if norm(N3)~=0
    resP=[N3(1,1),N3(2,1)]-RP;
    N=[RO_resO(1,1);RO_resO(1,2)];
else
    resO=tempresO;
end

%checking if the resP is within the allowable velocity sector for
%the phantom
function [check]=check_resP(resP,RP,gema)

tol=10^-5;
R1P_resP=acos(dot(resP,[RP(1,1),RP(2,1)])/(norm(resP)*norm(
```

```
[RP(1,1),RP(2,1)])));
% gema1
R2P_resP=acos(dot(resP,[RP(1,2),RP(2,2)])/(norm(resP)*norm(
[RP(1,2),RP(2,2)])));
% gema2
R3P_resP=acos(dot(resP,[RP(1,3),RP(2,3)])/(norm(resP)*norm(
[RP(1,3),RP(2,3)])));
% gema3
R4P_resP=acos(dot(resP,[RP(1,4),RP(2,4)])/(norm(resP)*norm(
[RP(1,4),RP(2,4)])));
% gema4

if R1P_resP>gema(1,1)+tol | R2P_resP>gema(1,2)+tol | R3P_resP>
    gema(1,3)+tol | R4P_resP>gema(1,4)+tol
    check=0;
    %resP does not lie within its allowable region
else
    check=1;
end


%decreasing the iteration time
function [N,resO,resP,v,gema,ECAVturnrate,turnrate]=
decrease_iteration_time(N,v,resP,resO,RO,RP,OHresO,ECAVturnrate,
PP2,RP_tot,RPt_tot,RO_tot,gema,turnrate,OHresP,within)
```

```
cont=1;
tempv=v;
temp_v=v(1,1);
tempresP=resP;
tempresO=resO;
tempgema=gema;
tempturnrate=turnrate;
tempECAVturnrate=ECAVturnrate;
lastresO=resO;
PH_turnradius=tempv(1,1)/(2*sin(turnrate/2));
ECAV_turnradius=tempv(1,4)/(2*sin(ECAVturnrate/2));

while norm(N)==0 & cont==1 & norm(v(1,1))>0.2*norm(temp_v)
    v=0.5*v;
    turnrate=2*asin(v(1,1)/(2*PH_turnradius));
    ECAVturnrate=2*asin(v(1,4)/(2*ECAV_turnradius));

    % get resP
    [gema]=get_gema(v,RO_tot,RP_tot);
    [Mu]=boundry_points(RP_tot,RPt_tot,gema,v);
    [V,in]=calresP(RP_tot,RPt_tot,PP2,gema,v);
    if in==1
        resP=V ;
    else
        [resP]=closestpoint(Mu,PP2);
    end
```

```
%checking if any of the ECAVs are within the range
[anyECAVwithin]=check_ECAV_within_range(RP_tot,RO_tot,v);

if anyECAVwithin==1 & OHresO~=1
    if ECAVturnrate < turnrate
        [resP]=check_tetaddot(resP,OHresP,ECAVturnrate-10^-4,
        RP_tot,gema);
    else
        [resP]=check_tetaddot(resP,OHresP,turnrate,RP_tot,
        gema);
    end
elseif OHresO~=1
    [resP]=check_tetaddot(resP,OHresP,turnrate,RP_tot,gema);
end

%get resO
[N1]=intpoints_resO(RO,RP,resP,v);
[N2]=upper_intpoints_resO(RO,RP,v,N1);
[N3]=updating_N_with_possible_better_intermediate_points(N1,
N2,resP,RP,RO,v,within,OHresO);
[N4]=updating_N_ECAV_turnrate_bounds(N3,RO,OHresO,
ECAVturnrate,v);
[N]=check_ECAV_turnrate(N4,RO,OHresO,ECAVturnrate);

if norm(N)~=0
    [resO]=better_intpoint_resO(RO,RP,resP,v,N);
```

```
        cont=0;
    else
        %choose the point that is closest to the turnrate bounds
        [resO]=point_closest_to_turnrate(RO,OHresO,N4);
    end

    alpha1=acos(dot(resO,OHresO)/(norm(resO)*norm(OHresO)));
    alpha2=acos(dot(lastresO,OHresO)/(norm(lastresO)*
    norm(OHresO)));

    if alpha1>=alpha2-0.1*ECAVturnrate
        'decreasing the iteration time doesn;t help'
        cont=0;
    else
        tempv=v;
        tempresP=resP;
        tempresO=resO;
        tempgema=gema;
        lastresO=resO;
    end
end
if norm(N)==0 & cont~=0
    'decreasing the iteration time helped but didn;t
    solve the problem'
end
```

```
if norm(N)==0
    v=tempv;
    resP=tempresP;
    resO=tempresO;
    gema=tempgema;
    turnrate=tempturnrate;
    ECAVturnrate=tempECAVturnrate;
end


%try increasing the iteration time
function [N,resO,resP,v,gema,ECAVturnrate,turnrate]=
increase_iteration_time(N,v,resP,resO,RO,RP,OHresO,ECAVturnrate,
PP2,RP_tot,RPt_tot,RO_tot,gema,turnrate,OHresP,within)

cont=1;
tempv=v;
temp_v=v(1,1);
tempresP=resP;
tempresO=resO;
tempgema=gema;
tempturnrate=turnrate;
tempECAVturnrate=ECAVturnrate;
lastresO=resO;
PH_turnradius=tempv(1,1)/(2*sin(turnrate/2));
ECAV_turnradius=tempv(1,4)/(2*sin(ECAVturnrate/2));
```

```
while norm(N)==0 & cont==1 & (norm(v(1,1)) < 4*norm(temp_v))
    v=2*v;
    turnrate=2*asin(v(1,1)/(2*PH_turnradius));
    ECAVturnrate=2*asin(v(1,4)/(2*ECAV_turnradius));
    % get resP
    [gema]=get_gema(v,RO_tot,RP_tot);
    [Mu]=boundry_points(RP_tot,RPt_tot,gema,v);
    [V,in]=calresP(RP_tot,RPt_tot,PP2,gema,v);
    if in==1
        resP=V ;
    else
        [resP]=closestpoint(Mu,PP2);
    end
    %checking if any of the ECAVs are within the range
    [anyECAVwithin]=check_ECAV_within_range(RP_tot,RO_tot,v);

    if anyECAVwithin==1 & OHresO~=1
        if ECAVturnrate < turnrate
            [resP]=check_tetaddot(resP,OHresP,ECAVturnrate-10^-4,
            RP_tot,gema);
        else
            [resP]=check_tetaddot(resP,OHresP,turnrate,RP_tot,
            gema);
        end
    elseif OHresO~=1
```

```
    [resP]=check_tetaddot(resP,OHresP,turnrate,RP_tot,gema);
end

%get resO
[N1]=intpoints_resO(RO,RP,resP,v);
[N2]=upper_intpoints_resO(RO,RP,v,N1);
[N3]=updating_N_with_possible_better_intermediate_points(N1,
N2,resP,RP,RO,v,within,OHresO);
[N4]=updating_N_ECAV_turnrate_bounds(N3,RO,OHresO,
ECAVturnrate,v);
[N]=check_ECAV_turnrate(N4,RO,OHresO,ECAVturnrate);

if norm(N)~=0
    [resO]=better_intpoint_resO(RO,RP,resP,v,N);
    cont=0;
else
    %choose the point that is closest to the turnrate bounds
    [resO]=point_closest_to_turnrate(RO,OHresO,N4);
end

alpha1=acos(dot(resO,OHresO)/(norm(resO)*norm(OHresO)));
alpha2=acos(dot(lastresO,OHresO)/(norm(lastresO)*
norm(OHresO)));
if alpha1>=alpha2-0.1*ECAVturnrate
    %increasing the iteration time doesn't help
    cont=0;
```

```
    else
        tempv=v;
        tempresP=resP;
        tempresO=resO;
        tempgema=gema;
        lastresO=resO;
    end
end
if norm(N)==0 & cont~=0
    'increasing the iteration time helped but didn;t
    solve the problem'
end
if norm(N)==0
    v=tempv;
    resP=tempresP;
    resO=tempresO;
    gema=tempgema;
    turnrate=tempturnrate;
    ECAVturnrate=tempECAVturnrate;
end


%recalculating resP and resO with reduced(halved) iteration time
'ecav#1----------'
[resO1,N1]=calresO(R1O1,R1P,tempresP,vmax,vomax,vomin,romin,
romax,OHresO1,ECAVturnrate,oldheading,turnrate);
```

```
'ecav#2-------------'
[resO2,N2]=calresO(R2O2,R2P,tempresP,vmax,vomax,vomin,romin,
romax,OHresO2,ECAVturnrate,oldheading,turnrate);
'ecav#3-----------'
[resO3,N3]=calresO(R3O3,R3P,tempresP,vmax,vomax,vomin,romin,
romax,OHresO3,ECAVturnrate,oldheading,turnrate);
'ecav#4---------------'
[resO4,N4]=calresO(R4O4,R4P,tempresP,vmax,vomax,vomin,romin,
romax,OHresO4,ECAVturnrate,oldheading,turnrate);



%plotting the trajectories of the phantom point and the UAVs
[m,n]=size(store_R1O2);
range1=[store_R1P,[R1R2(1,1);R1R2(1,2)],[R1R3(1,1);R1R3(1,2)],
    [R1R4(1,1);R1R4(1,2)],[R1P2(1,1);R1P2(1,2)],[0;0]];
a=max(range1(1,:));
b=max(range1(2,:));
c=min(range1(1,:));
d=min(range1(2,:));
if a<=0
    a=a-(a-c)*0.05;
else
    a=a+(a-c)*0.05;
end
c=c-(a-c)*0.05;
```

```
if b<=0
    b=b-(b-d)*0.05;
else
    b=b+(b-d)*0.05;
end
d=d-(b-d)*0.05;
hold on;
axis([c a d b]);
axis square ;
drawnow;
%pause(0.5);

x1=[store_R1P(1,1),store_R1P(1,1)];
y1=[store_R1P(2,1),store_R1P(2,1)];
h1=plot(x1,y1,'-b');
x2=[store_R1O1(1,1),store_R1O1(1,1)];
y2=[store_R1O1(2,1),store_R1O1(2,1)];
h2=plot(x2,y2,'-m');
x3=[store_R1O2(1,1),store_R1O2(1,1)];
y3=[store_R1O2(2,1),store_R1O2(2,1)];
h3=plot(x3,y3,'-r');
x4=[store_R1O3(1,1),store_R1O3(1,1)];
y4=[store_R1O3(2,1),store_R1O3(2,1)];
h4=plot(x4,y4,'-g');
x5=[store_R1O4(1,1),store_R1O4(1,1)];
y5=[store_R1O4(2,1),store_R1O4(2,1)];
```

```
h5=plot(x5,y5,'-k');
mov=avifile('animation2','fps',12);
frame=getframe;
mov=addframe(mov,frame);
legend([h1,h2,h3,h4,h5,h6,h7],'Phantom point','ECAV # 1',
'ECAV # 2','ECAV # 3','ECAV # 4','Radar points','Target point',0)
for i=1:n
    x1=[store_R1P(1,i),store_R1P(1,i)+store_resP(1,i)];
    y1=[store_R1P(2,i),store_R1P(2,i)+store_resP(2,i)];
    plot(x1,y1,'-b');
    %set(h1,'XData',x1,'YData',y1,'EraseMode','none');
    x2=[store_R1O1(1,i),store_R1O1(1,i)+store_resO1(1,i)];
    y2=[store_R1O1(2,i),store_R1O1(2,i)+store_resO1(2,i)];
    plot(x2,y2,'-m');
    %set(h2,'XData',x2,'YData',y2,'EraseMode','none');
    x3=[store_R1O2(1,i),store_R1O2(1,i)+store_resO2(1,i)];
    y3=[store_R1O2(2,i),store_R1O2(2,i)+store_resO2(2,i)];
    plot(x3,y3,'-r');
    x4=[store_R1O3(1,i),store_R1O3(1,i)+store_resO3(1,i)];
    y4=[store_R1O3(2,i),store_R1O3(2,i)+store_resO3(2,i)];
    plot(x4,y4,'-g');
    x5=[store_R1O4(1,i),store_R1O4(1,i)+store_resO4(1,i)];
    y5=[store_R1O4(2,i),store_R1O4(2,i)+store_resO4(2,i)];
    plot(x5,y5,'-k');
    %set(h3,'XData',x3,'YData',y3,'EraseMode','none');
    %pause(0.05);
```

```
     frame=getframe;
    mov=addframe(mov,frame);
end

legend([h1,h2,h3,h4,h5,h6,h7],'Phantom point','ECAV # 1',
'ECAV # 2','ECAV # 3','ECAV # 4','Radar points','Target point',0)
frame=getframe;
mov=addframe(mov,frame);
mov=close(mov);
```

VITA

Diyogu Hennadige Asanka Maithripala was born on March 3, 1976 in Sri Lanka. He received his Bachelor of Science in mechanical engineering from the University of Peradeniya, Sri Lanka in August 2001. In September 2003, he started his M.S. in Mechanical Engineering at Texas A&M University. He may be contacted through Dr. Suhada Jayasuriya at the Department of Mechanical Engineering, Texas A&M University, College Station, TX 77843-3123.