Games of Two Identical Vehicles

Ian Mitchell

July 2001

Stanford University
Department of Aeronautics and Astronautics Report
SUDAAR #740

The goal of this document is to flesh out and adapt the results of Merz in [1] to the three dimensional aircraft collision example. In both cases, the differential game scenario features two identical vehicles in a game of cat and mouse. The cat (called the pursuer below) is considered to capture the mouse (called the evader) if the cat ever gets within a certain distance—called the capture radius—of the mouse. Both vehicles are treated with kinematic models in which they have the same fixed forward velocity, but may choose their angular velocities within some range. The whole game is played in a three dimensional state space consisting of the relative position of the two vehicles in the plane and their relative heading.

We wish to compute points lying on the boundary of the reachable set, which is the boundary between those states from which the pursuer can cause a collision and those states from which the evader can escape. These points can then be used to test the accuracy of numerical Hamilton-Jacobi viscosity solutions of the same problem. For most of these points we can find explicit, analytic solutions; for a few we will need to use the numerical solution of an implicit pair of trigonometric equations. Note that the analysis below applies only when the two vehicles have identical models (speed and turn rates).

The notation from [1] is adopted, see table 1 and figure 1. The games we investigate use very generic dynamics for their vehicles; they would apply equally well as kinematic models for cars or aircraft:

$$\frac{d}{dt}\begin{bmatrix} x \\ y \\ \theta \end{bmatrix} = \begin{bmatrix} -\sigma_1 y + \sin\theta \\ -1 + \sigma_1 x + \cos\theta \\ -\sigma_1 + \sigma_2 \end{bmatrix}. \tag{1}$$

We will sometimes refer to the system state vector as $z \in \mathbb{R}^3$

$$z = \begin{bmatrix} x & y & \theta \end{bmatrix}^T. \tag{2}$$

In section 1 we examine the version of the game corresponding to Merz's "game of two identical cars", in which the pursuer is fixed at the origin and facing along the positive y axis. In section 2 we examine the dual problem, in which the evader is fixed at the origin

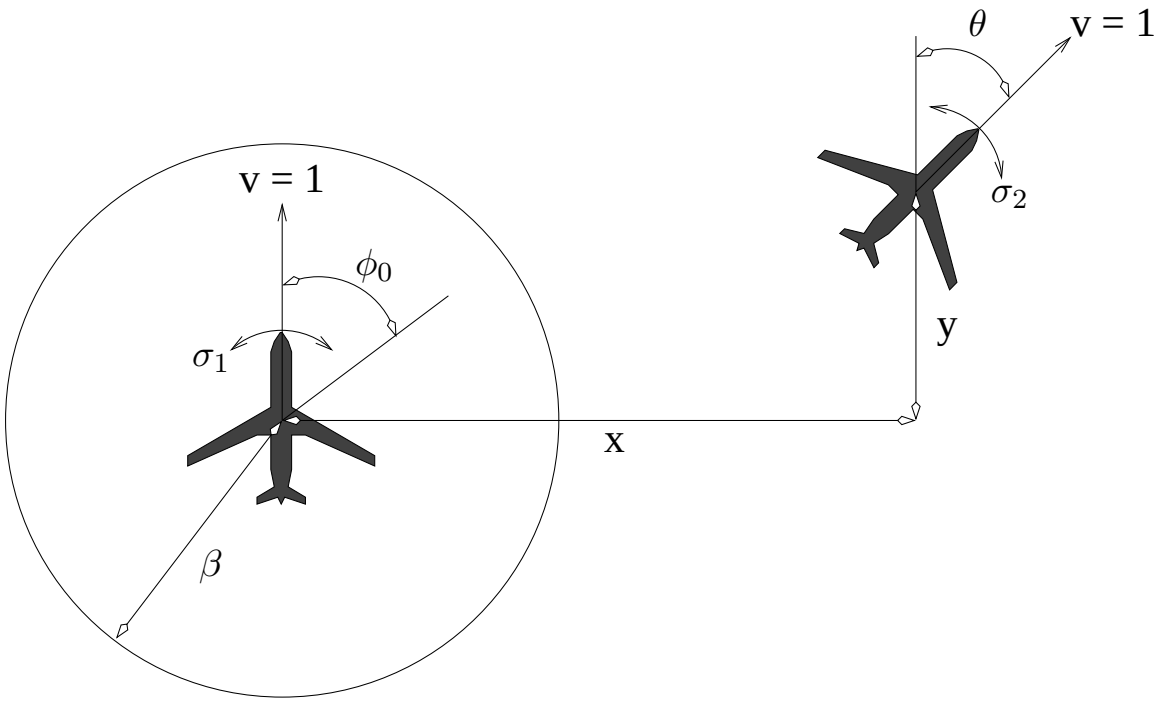

Figure 1: Coordinate system

and facing along the positive y axis—this game corresponds to what we have called the "aircraft collision avoidance" problem [3].

In both cases the inputs of both vehicles are restricted to the range ± 1. Merz recognizes that the absolute speed of the vehicles and the size of the capture circle do not matter, only their ratio β (Merz actually uses the minimum turn radius, but for the range of inputs available this turn radius is equal to the speed of the vehicles).

The next two sections detail the algorithms necessary to compute points on the barrier. Readers whose primary interest lies in validating a numerical Hamilton-Jacobi solver will want to skip directly to sections 3 and 4, which respectively describe the related Hamilton-Jacobi problem and the publicly available Matlab routines we have written to implement the results of sections 1 and 2 and thereby validate our own Hamilton-Jacobi solver.

1 Recreating the Game of Two Identical Cars

In this section we basically focus on reproducing Figure 7 from [1, p. 336], which studied the game with the pursuer at the origin. Many equations are exactly the same, except that we attempt to fill in algorithmic and algebraic steps to make the construction clear. We do not attempt explain why this construction produces the unsafe set—for those details, consult the original paper.

variable	meaning
x	relative position, perpendicular to flight direction of fixed aircraft
y	relative position, parallel to flight direction of fixed aircraft
θ	relative heading ($-\pi \leq \theta < \pi$)
σ_1	input of fixed aircraft ($\lvert\sigma_1\rvert \leq 1$)
σ_2	input of other aircraft ($\lvert\sigma_2\rvert \leq 1$)
β	ratio of capture radius to turn radius
ϕ_0	an angle measured counter-clockwise from the positive y-axis

Table 1: Variable definitions

1.1 The Optimal Trajectories

Merz claims that terminal conditions for all trajectories of interest can be written as

$$\begin{bmatrix} x(0) \\ y(0) \\ \theta(0) \end{bmatrix} = \begin{bmatrix} \beta \sin\phi_0 \\ \beta \cos\phi_0 \\ 2\phi_0 \end{bmatrix}, \tag{3}$$

for some ϕ_0. He further claims that trajectories leading from points on the boundary of the reachable set to these terminal conditions are made up of segments along which the inputs are fixed. Because the inputs are fixed, it is possible to find analytic equations for the trajectories in terms of their end conditions (x_0, y_0 and θ_0), the inputs (σ_1 and σ_2), and backwards time ($\tau = -t$ and terminal conditions become initial conditions):

Trajectory type 1: For inputs $\sigma_1 = -\sigma_2 = \pm 1$

$$\begin{bmatrix} x(\tau) \\ y(\tau) \\ \theta(\tau) \end{bmatrix} = \begin{bmatrix} x_0 \cos\tau + \sigma_1 \left[1 - \cos\tau + y_0 \sin\tau + \cos\theta - \cos(\theta_0 + \sigma_1\tau)\right] \\ y_0 \cos\tau + \sin\tau - \sigma_1 \left[x_0 \sin\tau + \sin\theta - \sin(\theta_0 + \sigma_1\tau)\right] \\ \theta_0 + 2\sigma_1\tau \end{bmatrix}. \tag{4}$$

Trajectory type 2: For inputs $\sigma_1 = \sigma_2 = \pm 1$

$$\begin{bmatrix} x(\tau) \\ y(\tau) \\ \theta(\tau) \end{bmatrix} = \begin{bmatrix} x_0 \cos\tau + \sigma_1 \left[1 - \cos\tau + y_0 \sin\tau + \cos(\theta_0 + \sigma_1\tau) - \cos\theta_0\right] \\ y_0 \cos\tau + \sin\tau - \sigma_1 \left[x_0 \sin\tau + \sin(\theta_0 + \sigma_1\tau) - \sin\theta_0\right] \\ \theta_0 \end{bmatrix}. \tag{5}$$

Trajectory type 3: For inputs $\sigma_1 = 0$ and $\sigma_2 = \pm 1$

$$\begin{bmatrix} x(\tau) \\ y(\tau) \\ \theta(\tau) \end{bmatrix} = \begin{bmatrix} x_0 - \sigma_2(\cos\theta - \cos\theta_0) \\ y_0 + \tau + \sigma_2(\sin\theta - \sin\theta_0) \\ \theta_0 - \sigma_2\tau \end{bmatrix}. \tag{6}$$

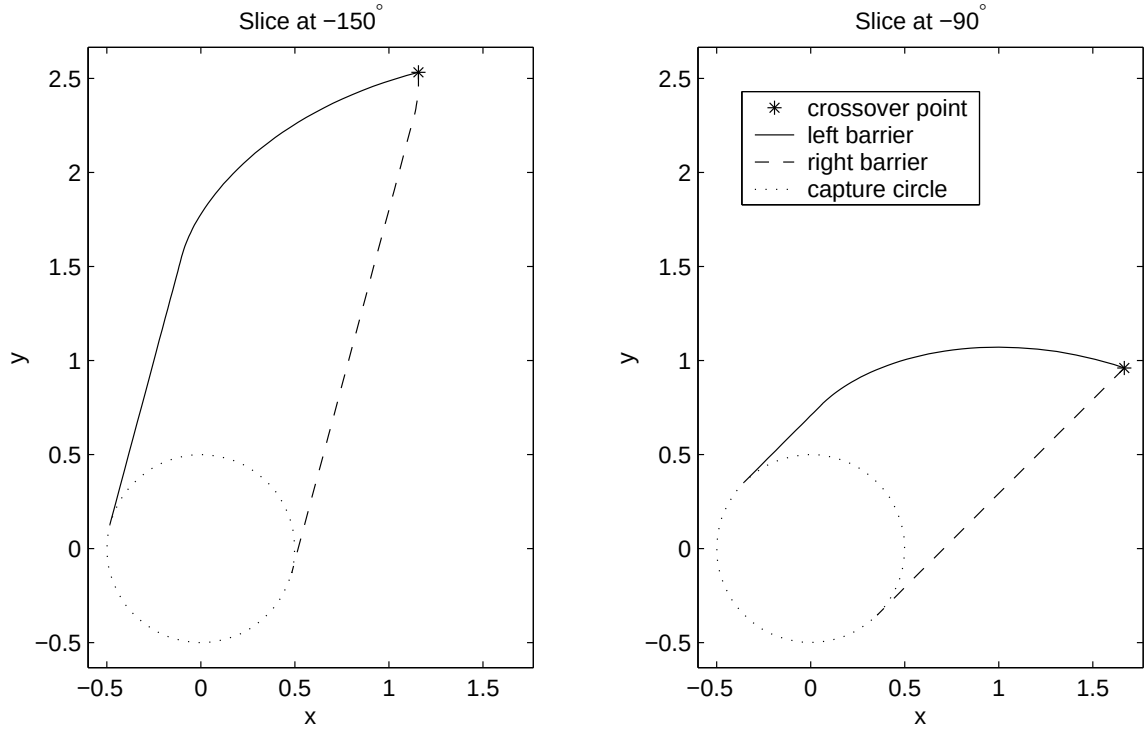

Figure 2: Barrier slices showing left and right barriers, the crossover point and the capture circle (for $\beta = 0.5$)

1.2 Determining the Boundary

Merz finds points on the boundary by integrating backward in time to those points from terminal points on the capture cylinder given by (3). The methods compute a slice of the barrier for fixed relative angle $\theta = \theta_2$[1] with the restriction $-\pi \leq \theta_2 \leq 0$; if necessary, the half of the barrier with $0 < \theta_2 < \pi$ can be determined by symmetry. It turns out that the slice of the reachable set at any θ_2 is bounded. This boundary consists of three curves: the first is some half of the capture circle, and the other two are smooth curves which start at the two ends of the half circle and meet at a sharp "crossover" point. We will call these latter two curves the left and right barriers—based on whether they lie to the left or right of a ray from the origin to the crossover point—and trajectories which lead from points on these barriers to the capture cylinder will be called left and right trajectories. Figure 2 shows two example barrier slices, labelling the left and right barriers. We will see later that trajectories which lead from the left barrier may sometimes appear to bear right and vice versa; however, we will label trajectories according to which barrier they touch and not according to the individual trajectory's apparent direction.

The difference between left and right will turn out to be the choice of the evader's input σ_2

[1] the motivation for choosing the index 2 for the parameter θ_2 will become clearer in the next section

throughout the trajectories; $\sigma_2 = +1$ will lead to left trajectories and barriers and $\sigma_2 = -1$ will lead to right trajectories and barriers. For this reason, the left barrier will be referred to as the positive barrier and quantities associated with it given the "+" superscript, while the right will be the negative and given the "−" superscript. To avoid confusion, it is important to remember that the "left" and "right" refer to the position of the barrier, and not to the direction that the evader is turning; in fact, $\sigma_2 = +1$ means that the evader is turning right, but it generates the left barrier.

The two barriers are not in general feasible trajectories of the system, but are made up of the initial conditions for trajectories which terminate on the capture cylinder. But in some cases (for θ_2 near zero) portions of the left barrier will turn out to be feasible trajectories.

In the discussion that follows, we have two slight differences with the notation in [1]. We use $\tau_2^{\pm}$ instead of $\tau^{\pm}$ and we have introduced the parameter θ_2 to avoid confusion between the state space variable θ and the parameter specifying which slice of the barrier we are computing—θ_2 is substituted for the latter.

1.3 The Crossover Point

The first step in computing points on the boundary at fixed θ_2 is to determine the crossover point where the barriers meet for that θ_2. This point is the end of two different backward time trajectories using $\sigma_2 = \pm 1$. Depending on the value of θ_2, these two trajectories may or may not lead to the same backwards time initial condition on the capture cylinder, but even if they do lead to the same initial condition they take distinct paths. There are four types of optimal trajectories leading from the capture cylinder to the crossover point, two associated with the left barrier (L1 and L2) and two with the right (R1 and R2):

Crossover Trajectory L1: This left trajectory starts (in backward time) at the point given by (3) with $\phi_0^+ = 0$ and consists of two segments, during both of which $\sigma_2 = +1$. The first segment is of type 3 and follows (6) for $\tau \in [0, \tau_1^+]$. The second segment is of type 1 and follows (4) for $\tau \in [\tau_1^+, \tau_2^+]$.

Crossover Trajectory R1: This right trajectory is the negative version of L1. It starts at the point given by (3) with $\phi_0^- = 0$ and consists of two segments, during both of which $\sigma_2 = -1$. The first segment is of type 3 and follows (6) for $\tau \in [0, \tau_1^-]$. The second segment is of type 1 and follows (4) for $\tau \in [\tau_1^-, \tau_2^-]$.

Crossover Trajectory L2: This left trajectory also starts at the point given by (3) with $\phi_0^+ = 0$ and consists of two segments, during both of which $\sigma_2 = +1$. The first segment is the same as the first segment of L1. The second segment, however, is of type 2, and follows (5) for $\tau \in [\tau_1^+, \tau_2^+]$.

Crossover Trajectory R2: This right trajectory is unlike any of the previous trajectories. It starts at the point given by (3) but with $\phi_0^- > 0$, and consists of only a single segment of type 1, which follows (4) for $\tau \in [0, \tau_2^-]$ (during which $\sigma_2 = -1$).

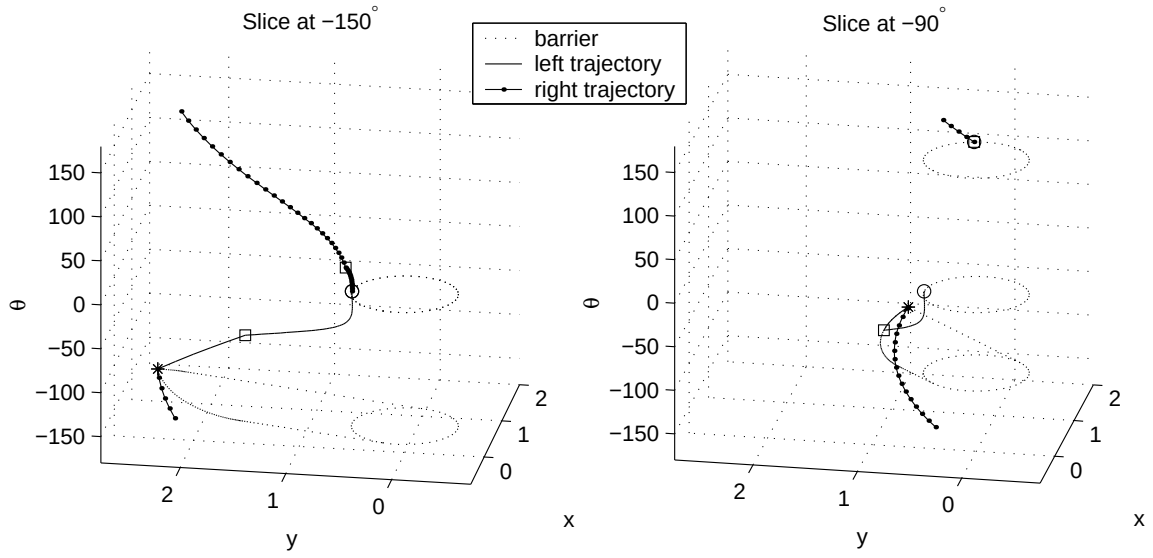

Figure 3: Barrier slices showing left and right crossover trajectories (for $\beta = 0.5$)

Figure 3 shows the left and right crossover trajectories for two sample barrier slices. In the left subplot, crossover trajectories L1 and R1 both start at the circle symbol, and finish at the crossover point labelled by a star. The box symbol on each trajectory shows where the trajectory switches segments. The right trajectory has a gap due to the periodic θ boundary conditions. The right subplot shows a slice where trajectories L2 and R2 occur. Note that the second segment of L2 lies on the barrier, and that R2 has only a single segment which starts with $\theta^-(0) > 0$. Several slices of the capture cylinder are shown to demonstrate that all trajectories start from it at some value of θ. For slices of the barrier at other values of θ_2, other combinations of L1, L2, R1, and R2 may occur.

For all these types of crossover trajectories, in order to find the barrier crossover point we construct trajectories

$$\begin{bmatrix} x^+(\tau) \\ y^+(\tau) \\ \theta^+(\tau) \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} x^-(\tau) \\ y^-(\tau) \\ \theta^-(\tau) \end{bmatrix}, \tag{7}$$

from the appropriate combination of (4), (5) and/or (6). The trajectories will be functions of the unknowns τ_2^+, τ_2^-, τ_1^+ and either τ_1^- or ϕ_0^- (a total of four unknowns). For crossover to occur at $\theta = \theta_2$, we know

$$\theta^+(\tau_2^+) = \theta_2 = \theta^-(\tau_2^-). \tag{8}$$

The crossover point is where these two trajectories meet

$$\begin{aligned} x^+(\tau_2^+) &= x^-(\tau_2^-), \\ y^+(\tau_2^+) &= y^-(\tau_2^-). \end{aligned} \tag{9}$$

The combination of (8) and (9) gives us four nonlinear equations for the four unknowns. After solving for $\tau_2^\pm$, the crossover point can be extracted from either side of (9).

The next four subsections detail the construction of the trajectories in (7) for each of the four cases L1, L2, R1 and R2. With that information, section 1.3.5 explains concretely how to determine the crossover point.

1.3.1 Crossover Trajectory L1

First, find the left trajectory's location after the first τ_1^+ time units from initial conditions ($\phi_0 = 0$), $\sigma_2 = +1$, and (6) (we start with $\theta^+(\tau_1^+)$ because it is needed for the other two coordinates' equations)

$$\begin{aligned}\theta^+(\tau_1^+) &= 2\phi_0 - \sigma_2\tau_1^+ \\ &= -\tau_1^+\end{aligned} \tag{10}$$

$$\begin{aligned}x^+(\tau_1^+) &= \beta\sin\phi_0 - \sigma_2(\cos\theta^+(\tau_1^+) - \cos 2\phi_0) \\ &= -\cos(-\tau_1^+) + 1 \\ &= -\cos\tau_1^+ + 1\end{aligned} \tag{11}$$

$$\begin{aligned}y^+(\tau_1^+) &= \beta\cos\phi_0 + \tau_1^+ + \sigma_2(\sin\theta^+(\tau_1^+) - \sin 2\phi_0) \\ &= \beta + \tau_1^+ + \sin(-\tau_1^+) \\ &= \beta + \tau_1^+ - \sin\tau_1^+\end{aligned} \tag{12}$$

For the subsequent $\tau_2^+ - \tau_1^-$ time units, the left trajectory follows (4) with $\sigma_2 = -\sigma_1 = +1$

$$\begin{aligned}\theta^+(\tau_2^+) &= \theta^+(\tau_1^+) + 2\sigma_1(\tau_2^+ - \tau_1^+), \\ &= \tau_1^+ - 2\tau_2^+.\end{aligned} \tag{13}$$

$$\begin{aligned}x^+(\tau_2^+) &= x^+(\tau_1^+)\cos(\tau_2^+ - \tau_1^+) + \sigma_1[1 - \cos(\tau_2^+ - \tau_1^+) + \cos\theta^+(\tau_2^+) \\ &\quad + y^+(\tau_1^+)\sin(\tau_2^+ - \tau_1^+) - \cos(\theta^+(\tau_1^+) + \sigma_1(\tau_2^+ - \tau_1^+))], \\ &= (1 - \cos\tau_1^+)\cos(\tau_2^+ - \tau_1^+) - [1 - \cos(\tau_2^+ - \tau_1^+) + \cos(\tau_1^+ - 2\tau_2^+) \\ &\quad - (\beta + \tau_1^+ - \sin(\tau_1^+))\sin(\tau_2^+ - \tau_1^+) - \cos(-\tau_2^+)], \\ &= -1 + 2\cos(\tau_2^+ - \tau_1^+) - \cos(2\tau_2^+ - \tau_1^+) - (\beta + \tau_1^+)\sin(\tau_2^+ - \tau_1^+) \\ &\quad + \cos\tau_2^+ - \cos\tau_1^+\cos(\tau_2^+ - \tau_1^+) - \sin\tau_1^+\sin(\tau_2^+ - \tau_1^+), \\ &= -1 + 2\cos(\tau_2^+ - \tau_1^+) - \cos(2\tau_2^+ - \tau_1^+) - (\beta + \tau_1^+)\sin(\tau_2^+ - \tau_1^+).\end{aligned} \tag{14}$$

$$\begin{aligned}y^+(\tau_2^+) &= y^+(\tau_1^+)\cos(\tau_2^+ - \tau_1^+) + \sin(\tau_2^+ - \tau_1^+) \\ &\quad - \sigma_1[\cos\theta^+(\tau_2^+) + x^+(\tau_1^+)\sin(\tau_2^+ - \tau_1^+) - \sin(\theta^+(\tau_1^+) + \sigma_1(\tau_2^+ - \tau_1^+))], \\ &= (\beta + \tau_1^+ - \sin(\tau_1^+))\cos(\tau_2^+ - \tau_1^+) + \sin(\tau_2^+ - \tau_1^+) \\ &\quad + [\cos(\tau_1^+ - 2\tau_2^+) + (1 - \cos\tau_1^+)\sin(\tau_2^+ - \tau_1^+) - \sin(-\tau_2^+)], \\ &= 2\sin(\tau_2^+ - \tau_1^+) - \sin(2\tau_2^+ - \tau_1^+) + (\beta + \tau_1^+)\cos(\tau_2^+ - \tau_1^+) \\ &\quad + \sin\tau_2^+ - \sin\tau_1^+\cos(\tau_2^+ - \tau_1^+) - \cos\tau_1^+\cos(\tau_2^+ - \tau_1^+), \\ &= 2\sin(\tau_2^+ - \tau_1^+) - \sin(2\tau_2^+ - \tau_1^+) + (\beta + \tau_1^+)\cos(\tau_2^+ - \tau_1^+).\end{aligned} \tag{15}$$

The last simplification in both the x^+ and y^+ sequences is possible because of the trigonometric identities

$$\cos\tau_1\cos(\tau_2-\tau_1)+\sin\tau_1\sin(\tau_2-\tau_1)=\cos\tau_2,$$
$$\sin\tau_1\cos(\tau_2-\tau_1)+\cos\tau_1\cos(\tau_2-\tau_1)=\sin\tau_2.$$

From (8) and (13) we find

$$\tau_1^+=\theta_2+2\tau_2^+. \tag{16}$$

This equation can be used to remove the unknown τ_1^+ from the equations (14) and (15)

$$\begin{aligned} x^+(\tau_2^+) &= -1+2\cos(\tau_2^+-\theta_2-2\tau_2^+)-\cos(2\tau_2^+-\theta_2+2\tau_2^+) \\ &\quad -(\beta+\theta_2+2\tau_2^+)\sin(\tau_2^+-\theta-2\tau_2^+) \qquad (17)\\ &= -1+2\cos(\theta_2+\tau_2^+)-\cos\theta_2+(\beta+\theta_2+2\tau_2^+)\sin(\theta_2+\tau_2^+) \\ y^+(\tau_2^+) &= 2\sin(\tau_2^+-\theta_2-2\tau_2^+)-\sin(2\tau_2^+-\theta_2-2\tau_2^+) \\ &\quad +(\beta+\theta_2+2\tau_2^+)\cos(\tau_2^+-\theta_2-2\tau_2^+) \qquad (18)\\ &= -2\sin(\theta_2+\tau_2^+)+\sin\theta_2+(\beta+\theta_2+2\tau_2^+)\cos(\theta_2+\tau_2^+) \end{aligned}$$

1.3.2 Crossover Trajectory R1

Following a similar procedure for the right trajectory (with $\sigma_2=-\sigma_1=-1$) we find

$$\begin{aligned} \theta^-(\tau_1^-) &= 2\phi_0-\sigma_2\tau_1^-, \\ &= \tau_1^-. \end{aligned} \tag{19}$$

$$\begin{aligned} x^-(\tau_1^-) &= \beta\sin\phi_0-\sigma_2(\cos\theta^-(\tau_1^-)-\cos 2\phi_0), \\ &= \cos(\tau_1^-)-1. \end{aligned} \tag{20}$$

$$\begin{aligned} y^-(\tau_1^-) &= \beta\cos\phi_0+\tau_1^-+\sigma_2(\sin\theta^-(\tau_1^-)-\sin 2\phi_0), \\ &= \beta+\tau_1^--\sin(\tau_1^-). \end{aligned} \tag{21}$$

and

$$\begin{aligned} \theta^-(\tau_2^-) &= \theta^-(\tau_1^-)+2\sigma_1(\tau_2^--\tau_1^-), \\ &= -\tau_1^-+2\tau_2^-. \end{aligned} \tag{22}$$

$$\begin{aligned} x^-(\tau_2^-) &= x^-(\tau_1^-)\cos(\tau_2^--\tau_1^-)+\sigma_1[1-\cos(\tau_2^--\tau_1^-)+\cos\theta^-(\tau_2^-) \\ &\quad +y^-(\tau_1^-)\sin(\tau_2^--\tau_1^-)-\cos(\theta^-(\tau_1^-)+\sigma_1(\tau_2^--\tau_1^-))], \\ &= 1-2\cos(\tau_2^--\tau_1^-)+\cos(2\tau_2^--\tau_1^-)+(\beta+\tau_1^-)\sin(\tau_2^--\tau_1^-) \\ &\quad -\cos\tau_2^-+\cos\tau_1^-\cos(\tau_2^--\tau_1^-)-\sin\tau_1^-\sin(\tau_2^--\tau_1^-), \\ &= 1-2\cos(\tau_2^--\tau_1^-)+\cos(2\tau_2^--\tau_1^-)+(\beta+\tau_1^-)\sin(\tau_2^--\tau_1^-). \end{aligned} \tag{23}$$

$$\begin{aligned} y^-(\tau_2^-) &= y^-(\tau_1^-)\cos(\tau_2^--\tau_1^-)+\sin(\tau_2^--\tau_1^-) \\ &\quad -\sigma_1[\sin\theta^-(\tau_2^-)+x^-(\tau_1^-)\sin(\tau_2^--\tau_1^-)-\sin(\theta^-(\tau_1^-)+\sigma_1(\tau_2^--\tau_1^-))], \\ &= 2\sin(\tau_2^--\tau_1^-)-\sin(2\tau_2^--\tau_1^-)+(\beta+\tau_1^-)\cos(\tau_2^--\tau_1^-) \\ &\quad +\sin\tau_2^--\sin\tau_1^-\cos(\tau_2^--\tau_1^-)-\cos\tau_1^-\sin(\tau_2^--\tau_1^-), \\ &= 2\sin(\tau_2^--\tau_1^-)-\sin(2\tau_2^--\tau_1^-)+(\beta+\tau_1^-)\cos(\tau_2^--\tau_1^-). \end{aligned} \tag{24}$$

Again, the last three terms in the penultimate lines of (23) and (24) happen to cancel by trigonometric identities. Now use (8) and (22) to find

$$\tau_1^- = 2\tau_2^- - \theta_2, \tag{25}$$

and use this to eliminate τ_1^- from (23) and (24)

$$\begin{aligned} x^-(\tau_2^-) &= 1 - 2\cos(\tau_2^- - 2\tau_2^- + \theta_2) + \cos(2\tau_2^- - 2\tau_2^- + \theta_2) \\ &\quad + (\beta + 2\tau_2^- - \theta_2)\sin(\tau_2^- - 2\tau_2^- + \theta_2), \qquad (26) \\ &= 1 - 2\cos(\theta_2 - \tau_2^-) + \cos\theta_2 + (\beta + 2\tau_2^- - \theta_2)\sin(\theta_2 - \tau_2^-). \\ y^-(\tau_2^-) &= 2\sin(\tau_2^- - 2\tau_2^- + \theta_2) - \sin(2\tau_2^- - 2\tau_2^- + \theta_2) \\ &\quad + (\beta + 2\tau_2^- - \theta_2)\cos(\tau_2^- - 2\tau_2^- + \theta_2), \qquad (27) \\ &= 2\sin(\theta_2 - \tau_2^-) - \sin\theta_2 + (\beta + 2\tau_2^- - \theta_2)\cos(\theta_2 - \tau_2^-). \end{aligned}$$

In regard to (26) and (27), there is one final detail. Remember that $\theta_2 \in [0, -\pi]$. Looking at trajectories generated by (4) and (6) with $\sigma_2 = -\sigma_1 = -1$, however, we see that $\theta^-(\tau)$ is increasing as τ increases. The periodicity of the θ coordinate is automatically handled whenever it appears inside a trigonometric function, but while constructing (26) and (27) we have in places substituted θ_2 for a value of $\tau > 0$. In those places we have to manually take account of the range of θ_2 and periodicity by substituting $2\pi + \theta_2$ to get

$$x^-(\tau_2^-) = 1 - 2\cos(\theta_2 - \tau_2^-) + \cos\theta_2 + (\beta + 2\tau_2^- - 2\pi - \theta_2)\sin(\theta_2 - \tau_2^-). \tag{28}$$

$$y^-(\tau_2^-) = 2\sin(\theta_2 - \tau_2^-) - \sin\theta_2 + (\beta + 2\tau_2^- - 2\pi - \theta_2)\cos(\theta_2 - \tau_2^-). \tag{29}$$

1.3.3 Crossover Trajectory L2

The optimal segments of this left trajectory still follow (6) for $\tau \in [0, \tau_1^+]$, but then switch to (5) for $\tau \in [\tau_1^+, \tau_2^+]$. Therefore, we can use (10), (11) and (12) for the initial segment to τ_1^+. Then from (5)

$$\theta^+(\tau_2^+) = \theta^+(\tau_1^+) = -\tau_1^+ \tag{30}$$

and from (8)

$$\tau_1^+ = -\theta_2. \tag{31}$$

Given this equation and (5) the crossover point for this trajectory is

$$\begin{aligned}
x^+(\tau_2^+) &= x^+(\tau_1^+)\cos(\tau_2^+ - \tau_1^+) + \sigma_1[1 - \cos(\tau_2^+ - \tau_1^+) \\
&\quad + y^+(\tau_1^+)\sin(\tau_2^+ - \tau_1^+) + \cos(\theta^+(\tau_1^+) + \sigma_1(\tau_2^+ - \tau_1^+)) - \cos\theta^+(\tau_1^+)], \\
&= (1 - \cos\tau_1^+)\cos(\tau_2^+ - \tau_1^+) + 1 - \cos(\tau_2^+ - \tau_1^+) \\
&\quad + (\beta + \tau_1^+ - \sin\tau_1^+)\sin(\tau_2^+ - \tau_1^+) + \cos(\theta_2 + \tau_2^+ - \tau_1^+) - \cos\theta_2, \\
&= 1 - \cos\theta_2[1 + \cos(\tau_2^+ + \theta_2)] \\
&\quad + (\beta - \theta_2 + \sin\theta_2)\sin(\tau_2^+ + \theta_2) + \cos(\tau_2^+ + 2\theta_2), \\
&= 1 - \cos\theta_2 + (\beta - \theta_2)\sin(\tau_2^+ + \theta_2).
\end{aligned} \tag{32}$$

$$\begin{aligned}
y^+(\tau_2^+) &= y^+(\tau_1^+)\cos(\tau_2^+ - \tau_1^+) + \sin(\tau_2^+ - \tau_1^+) \\
&\quad - \sigma_1[x^+(\tau_1^+)\sin(\tau_2^+ - \tau_1^+) + \sin(\theta^+(\tau_1^+) + \sigma_1(\tau_2^+ - \tau_1^+)) - \sin\theta^+(\tau_1^+)], \\
&= (\beta + \tau_1^+ - \sin\tau_1^+)\cos(\tau_2^+ - \tau_1^+) + \sin(\tau_2^+ - \tau_1^+) \\
&\quad - (1 - \cos\tau_1^+)\sin(\tau_2^+ - \tau_1^+) - \sin(\theta_2 + \tau_2^+ - \tau_1^+) + \sin\theta_2, \\
&= (\beta - \theta_2 + \sin\theta_2)\cos(\tau_2^+ + \theta_2) + \cos\theta_2\sin(\tau_2^+ + \theta_2) \\
&\quad + \sin\theta_2 - \sin(\tau_2^+ + 2\theta_2), \\
&= \sin\theta_2 + (\beta - \theta_2)\cos(\tau_2^+ + \theta_2).
\end{aligned} \tag{33}$$

We have used the useful trigonometric identities

$$\begin{aligned}
\sin\theta_2\sin(\tau + \theta_2) - \cos\theta_2\cos(\tau + \theta_2) &= -\cos(\tau + 2\theta_2), \\
\sin\theta_2\cos(\tau + \theta_2) + \cos\theta_2\sin(\tau + \theta_2) &= \sin(\tau + 2\theta_2),
\end{aligned}$$

in determining x^- and y^-.

1.3.4 Crossover Trajectory R2

For this right trajectory, the initial conditions are $\phi_0^- \neq 0$, and (4) is followed for $\tau \in [0, \tau_2^-]$. We start with (4)

$$\begin{aligned}
\theta^-(\tau_2^-) &= 2\phi_0^- + 2\sigma_1\tau_2^-, \\
&= 2\phi_0^- + 2\tau_2^-,
\end{aligned} \tag{34}$$

so that (8) gives us

$$\phi_0^- = \frac{\theta_2 - 2\tau_2^-}{2}. \tag{35}$$

Starting again at (4) we can determine the right trajectory's position at crossover as

$$\begin{aligned} x^-(\tau_2^-) &= \beta\sin\phi_0^-\cos\tau_2^- + \sigma_1[1-\cos\tau_2^- + \beta\cos\phi_0^-\sin\tau_2^- \\ &\qquad + \cos\theta^-(\tau_2^-) - \cos(2\phi_0^- + \sigma_1\tau_2^-)], \\ &= 1-\cos\tau_2^- + \beta\sin(\phi_0^- + \tau_2^-) - \cos(2\phi_0^- + \tau_2^-) + \cos\theta_2, \\ &= 1-\cos\tau_2^- + \beta\sin\tfrac{2\pi+\theta_2}{2} - \cos(\theta_2 - \tau_2^-) + \cos\theta_2. \end{aligned} \tag{36}$$

$$\begin{aligned} y^-(\tau_2^-) &= \beta\cos\phi_0^-\cos\tau_2^- + \sin\tau_2^- \\ &\qquad - \sigma_1[\beta\sin\phi_0^-\sin\tau_2^- + \sin\theta^-(\tau_2^-) - \sin(2\phi_0^- + \sigma_1\tau_2^-)], \\ &= \sin\tau_2^- + \beta\cos(\phi_0^- + \tau_2^-) + \sin(2\phi_0^- + \tau_2^-) - \sin\theta_2, \\ &= \sin\tau_2^- + \beta\cos\tfrac{2\pi+\theta_2}{2} + \sin(\theta_2 - \tau_2^-) - \sin\theta_2. \end{aligned} \tag{37}$$

We have used the useful trigonometric identities

$$\begin{aligned} \beta\sin\phi\cos\tau + \beta\cos\phi\sin\tau &= \beta\sin(\phi+\tau), \\ \beta\cos\phi\cos\tau - \beta\sin\phi\sin\tau &= \beta\cos(\phi+\tau), \end{aligned}$$

in determining x^- and y^-. We have also added 2π when θ_2 appears in a function which is not 2π periodic.

1.3.5 Crossover Computation

Depending on the values of θ_2 and β, either of L1 or L2 can meet either of R1 or R2 at the crossover point. Merz does not mention how he determines which of the four possible cases hold. We have adopted the trial and error approach—try a combination and see if its associated parameters are physically appropriate:

$$\begin{aligned} &\tau_1^+ \le \tau_2^+, \\ &\tau_1^- \ge 0, \\ &\phi_0^- \ge 0. \end{aligned} \tag{38}$$

If not, try another combination.

For example, we generally start with the assumption that L1 and R1 meet at the crossover point. Therefore, equate (17) with (28) and (18) with (29) to get two nonlinear equations in the two unknowns τ_2^+ and τ_2^- (θ_2 is given). Solve them in some manner (we use Matlab's `fsolve`). Then use (16) and (25) to determine τ_1^+ and τ_1^- respectively. If (38) holds, then the assumption of L1 and R1 was correct; proceed to section 1.4 to find the rest of the barrier for this θ_2.

Otherwise, try the other combinations until one is found that satisfies (38) (in our experience, one is always found)

- For L1 with R2, equate (17) with (36) and (18) with (37) to solve for τ_2^+ and τ_2^-. Then (16) and (35) can be used to find τ_1^+ and ϕ_0^-.

- For L2 with R1, equate (32) with (28) and (33) with (29) to solve for τ_2^+ and τ_2^-. Then (31) and (25) can be used to find τ_1^+ and τ_1^-.
- For L2 with R2, equate (32) with (36) and (33) with (37) to solve for τ_2^+ and τ_2^-. Then (31) and (35) can be used to find τ_1^+ and ϕ_0^-.

In all these cases, the crossover point is given by $x^+(\tau_2^+)$ and $y^+(\tau_2^+)$.

1.4 The Rest of the Barrier

Once the crossover point of the two barriers is found, the other points on the left and right barriers may be extracted. The scheme for computing points on a barrier depends on the type of trajectory that lead to the crossover point for that side of the barrier. We examine the construction at a fixed $\theta = \theta_2$ of each of the barrier types separately.

1.4.1 Barrier Type L1

Points on this type of barrier are the ends of backwards time trajectories. There are two subsets of points on this barrier, which we will call L1.1 and L1.2; the procedure for producing points in each subset is different.

Let L1.1 be the subset closer to the capture circle. Each point in L1.1 is the end of a different single segment backwards time trajectory. To construct one of these trajectories

1. Choose $\zeta \in [\frac{\theta_2}{2}, 0]$ and let $\eta = \frac{2\zeta - \theta_2}{2}$. These equations come from the θ portion of (4) with $\theta_0 = 2\zeta$ and $\sigma_2 = -\sigma_1 = +1$.
2. Use (4) to solve for $x(\eta)$ and $y(\eta)$, where x_0 and y_0 are given by (3) with $\phi_0 = \zeta$, and $\sigma_2 = -\sigma_1 = +1$.

The point in L1.1 is given by $\begin{bmatrix} x(\eta) & y(\eta) & \theta_2 \end{bmatrix}^T$.

L1.2 is the subset of the barrier closer to the crossover point. Each point in L1.2 is the end of a different two segment backwards time trajectory. To construct one of these trajectories

1. Choose $\eta_1 \in [0, \tau_1^+]$ and use (6) to solve for $x(\eta_1)$, $y(\eta_1)$ and $\theta(\eta_1)$, where x_0, y_0 and θ_0 are given by (3) with $\phi_0 = 0$, and $\sigma_2 = +1$.
2. Let $\eta_2 = \frac{\eta_1 - \theta_2}{2}$. This equation comes from the θ portion of (4) with $\theta_0 = \theta(\eta_1)$ and $\sigma_2 = -\sigma_1 = +1$.
3. Use (4) to solve for $x(\eta_2)$ and $y(\eta_2)$ with $x_0 = x(\eta_1)$, $y_0 = y(\eta_1)$ and $\sigma_2 = -\sigma_1 = +1$.

The point in L1.2 is given by $\begin{bmatrix} x(\eta_2) & y(\eta_2) & \theta_2 \end{bmatrix}^T$.

To construct an approximation of the whole barrier, we solve for a large number of points in each subset L1.1 and L1.2. Figure 4 shows overhead and oblique views of some of the

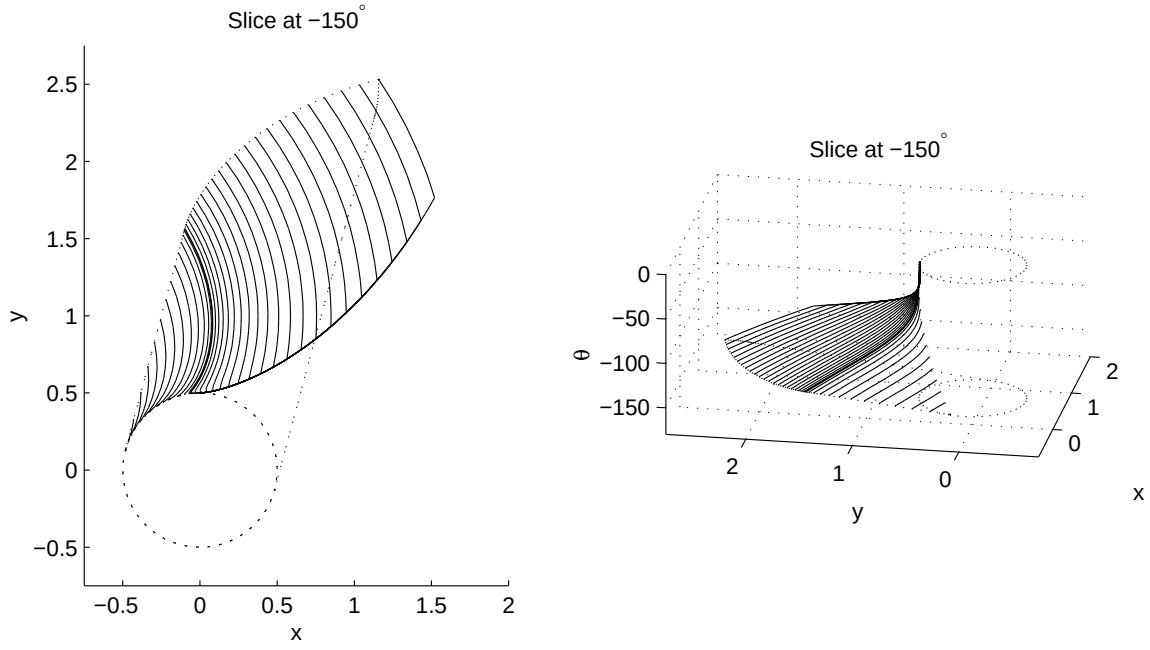

Figure 4: Two views of the trajectories whose endpoints form a slice of the left barrier (for $\beta = 0.5$)

trajectories which end on a barrier of type L1 for a particular value of θ_2. Subset L1.1 are the leftmost solid lines on the overhead view; in the oblique view these trajectories start from a variety of different initial conditions (based on the choice of ζ) on the side of the capture cylinder. Subset L1.2 trajectories all start from the point $\begin{bmatrix} 0 & \beta & 0 \end{bmatrix}^T$ and share an initial segment; the choice of η_1 determines when they branch from this segment.

1.4.2 Barrier Type R1

This barrier is the negative version of L1—it has two subsets R1.1 and R1.2, and each point in these subsets is the end point of a different backwards time trajectory.

For R1.1

1. Choose $\zeta \in [0, \frac{2\pi+\theta_2}{2}]$ and let $\eta = \frac{2\pi+\theta_2-2\zeta}{2}$ (from (4) with $\sigma_2 = -\sigma_1 = -1$). The 2π handles the periodic boundary conditions on θ.

2. Use (4) to solve for $x(\eta)$ and $y(\eta)$, where x_0 and y_0 are given by (3) with $\phi_0 = \zeta$, and $\sigma_2 = -\sigma_1 = -1$.

The point in R1.1 is given by $\begin{bmatrix} x(\eta) & y(\eta) & \theta_2 \end{bmatrix}^T$.

For R1.2

1. Choose $\eta_1 \in [0, \tau_1^-]$ and use (6) to solve for $x(\eta_1)$, $y(\eta_1)$ and $\theta(\eta_1)$, where x_0, y_0 and θ_0 are given by (3) with $\phi_0 = 0$, and $\sigma_2 = -1$.
2. Let $\eta_2 = \frac{\eta_1 + 2\pi + \theta_2}{2}$ (from (4) with $\sigma_2 = -\sigma_1 = -1$).
3. Use (4) to solve for $x(\eta_2)$ and $y(\eta_2)$ with $x_0 = x(\eta_1)$, $y_0 = y(\eta_1)$ and $\sigma_2 = -\sigma_1 = -1$.

The point in L1.2 is given by $\begin{bmatrix} x(\eta_2) & y(\eta_2) & \theta_2 \end{bmatrix}^T$.

1.4.3 Barrier Type L2

This barrier type includes three subsets L2.1, L2.2 and L2.3. The first two are constructed in exactly the same fashion as subsets L1.1 and L1.2 (respectively) from section 1.4.1. The subset L2.3—which lies closer to the crossover point than L2.2—consists of a single type 2 trajectory constructed by

1. Compute $x(\tau_1^+)$ and $y(\tau_1^+)$ from (6) with x_0 and y_0 from (3) (with $\phi_0 = 0$) and $\sigma_2 = +1$ (it is also possible to compute $\theta(\tau_1^+)$, but it will turn out to be θ_2).
2. Compute $x(\tau)$ and $y(\tau)$ from (5) for $\tau \in [\tau_1^+, \tau_2^+]$ with $x_0 = x(\tau_1^+)$, $y_0 = y(\tau_1^+)$, and $\sigma_2 = \sigma_1 = +1$.

L2.3 is this segment of trajectory $x(\tau)$ and $y(\tau)$ for $\tau \in [\tau_1^+, \tau_2^+]$.

1.4.4 Barrier Type R2

The single subset R2.1 of this barrier is constructed in the same manner as R1.1 from section 1.4.2, except for a restricted range $\zeta \in [\phi_0^-, \frac{2\pi + \theta_2}{2}]$.

1.5 Results

Figure 5 shows cross sections of the barrier for $\beta = 0.5$ with the pursuer at the origin. On the left is an $x - y$ view superimposing the sections, while on the right the sections are shown at their appropriate θ values in three dimensions. Sections are shown at $\theta_2 = -180°, -150°, -120°, -90°, -60°, -30°$ and $0°$. The $\theta_2 = 0°$ section is difficult to see on the left because it is precisely the capture circle. The crossover point for each section is marked with a star.

The sections at $\theta_2 = -180°$ and $\theta_2 = -150°$ consist of barriers L1 and R1, while the remaining sections are made up of barriers L2 and R2. For this particular choice of β, L2 and R1 never appear together, while L1 and R2 occur together only for $\theta_2 \in [-143°, -123°]$ and so none of the chosen sections are of this type.

On the upper two cross sections' barriers, the straight portions between circle and triangle are L1.1 and R1.1 on left and right respectively. The curved portions between triangle and star are L1.2 and R1.2 respectively. On the remaining sections, R2.1 is the entire

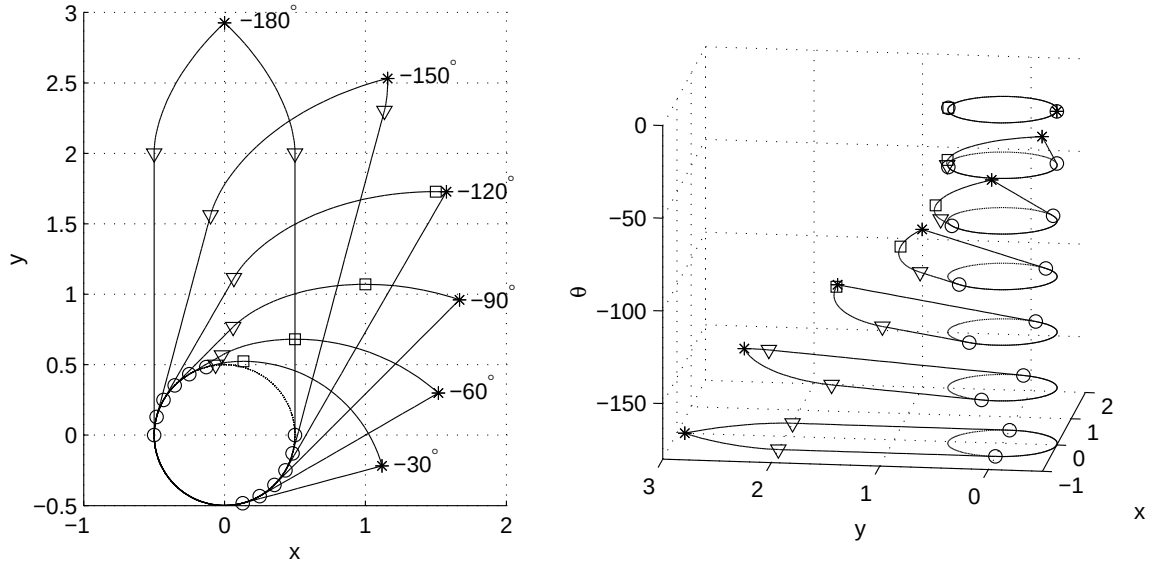

Figure 5: Cross sections of the barrier for $\beta = 0.5$ and the pursuer at the origin (left subplot is a recreation of figure 7 from [1])

right barrier (between circle and star). L2.1 lies between circle and triangle, L2.2 between triangle and square, and L2.3 between square and star.

2 Adaptation to Aircraft Collision Avoidance

For the two vehicle game described in [2, 3], the evader (control) lies at the origin and is surrounded by the capture circle. In the relative coordinate system dynamics (1), the input σ_1 is attached not to the pursuer, but to the vehicle at the origin (and conversely for σ_2); consequently, to compute results for the aircraft game we must swap the inputs. Unfortunately, while this change of inputs does not modify the absolute motions of the game players, it causes subtle but important changes in the mapping of those motions into the relative coordinate system. The result produces slightly different equations for the optimal trajectories and thus a slightly different unsafe set.

This section analyzes the changes which must be made to the equations from the previous section in order to place the evader at the origin instead of the pursuer. During our analysis we swap the inputs—now the evader uses input σ_1 and the pursuer σ_2—but do not make the other changes necessary to convert fully to the example presented in [2] (except for figure 7 at the end of section 2.5; we explain the other necessary changes in the figure's accompanying text).

2.1 Optimal Trajectories

The same terminal conditions apply, but now the coordinate system is relative to the evader at the origin. Therefore, (3) changes to

$$\begin{bmatrix} x(0) \\ y(0) \\ \theta(0) \end{bmatrix} = \begin{bmatrix} -\beta \sin \phi_0 \\ -\beta \cos \phi_0 \\ 2\phi_0 \end{bmatrix}, \tag{39}$$

The inputs were symmetric for trajectories of types 1 and 2, so (4) and (5) continue to apply. The asymmetrical inputs for trajectories of type 3, however, require a change to its governing equations. In our modified problem, the pursuer still chooses not to turn, but is now using σ_2 as its input. Therefore we have $\sigma_1 = \pm 1$, $\sigma_2 = 0$ and instead of (6)

$$\begin{bmatrix} x(\tau) \\ y(\tau) \\ \theta(\tau) \end{bmatrix} = \begin{bmatrix} x_0 \cos \tau - \tau \sin \theta + \sigma_1 [1 - \cos \tau + y_0 \sin \tau] \\ y_0 \cos \tau - \tau \cos \theta + \sin \tau - \sigma_1 x_0 \sin \tau \\ \theta_0 + \sigma_1 \tau \end{bmatrix}, \tag{40}$$

which were determined with some help from Matlab's `dsolve` and the observation that $\sigma_1^2 = 1$.

2.2 Determining the Boundary

The left barrier first described in section 1.2 is generated by trajectories in which the evader's input is positive and quantities associated with it are given the "+" superscript, while the right barrier comes from trajectories in which the evader's input is negative and are given the "−" superscript. We will keep these designations, but must make note of the fact that in the new game the evader's input is σ_1, and so the superscripts will agree with the value of σ_1 and not that of σ_2.

2.3 The Crossover Points

The general procedure is the same as that in section 1.3. Once again there are two types of left trajectories L1 and L2 leading to crossover, and two types of right trajectories R1 and R2. It turns out that L1 with the evader at the origin has a structure very similar to L1 with the pursuer at the origin from section 1.3.1, while evader R1 looks like pursuer R1 from section 1.3.2. On the other hand, evader L2 turns out to be like pursuer R2 from section 1.3.4 and evader R2 like pursuer L2 from section 1.3.3. These are, however, structural similarities; in most cases the details of the equations are quite different.

2.3.1 Crossover Trajectory L1

We use (40) to replace (10), (11) and (12) (remembering that now $\sigma_1 = +1$ on the left)

$$\begin{aligned}\theta^+(\tau_1^+) &= 2\phi_0 + \sigma_1\tau_1^+ \\ &= \tau_1^+\end{aligned} \tag{41}$$

$$\begin{aligned}x^+(\tau_1^+) &= -\beta\sin\phi_0\cos\tau_1^+ - \tau_1^+\sin\theta^+(\tau_1^+) + \sigma_1[1 - \beta\cos\phi_0\sin\tau_1^+ - \cos\tau_1^+], \\ &= -\tau_1^+\sin\tau_1^+ + 1 - \beta\sin\tau_1^+ - \cos\tau_1^+, \\ &= 1 - \cos\tau_1^+ - (\beta + \tau_1^+)\sin\tau_1^+.\end{aligned} \tag{42}$$

$$\begin{aligned}y^+(\tau_1^+) &= -\beta\cos\phi_0\cos\tau_1^+ - \tau_1^+\cos\theta^+(\tau_1^+) + \sin\tau_1^+ + \sigma_1\beta\sin\phi_0\sin\tau_1^+, \\ &= -\beta\cos\tau_1^+ - \tau_1^+\cos\tau_1^+ + \sin\tau_1^+, \\ &= \sin\tau_1^+ - (\beta + \tau_1^+)\cos\tau_1^+.\end{aligned} \tag{43}$$

Then use (4) with (41) to get a replacement for (13)

$$\begin{aligned}\theta^+(\tau_2^+) &= \theta^+(\tau_1^+) + 2\sigma_1(\tau_2^+ - \tau_1^+), \\ &= 2\tau_2^+ - \tau_1^+.\end{aligned} \tag{44}$$

and then with (8) to get a replacement for (16)

$$\tau_1^+ = 2\tau_2^+ - \theta_2. \tag{45}$$

Now start again with (4) and using (42), (43) and (45) get replacements for (17) and (18)

$$\begin{aligned}x^+(\tau_2^+) &= x^+(\tau_1^+)\cos(\tau_2^+ - \tau_1^+) + \sigma_1[1 - \cos(\tau_2^+ - \tau_1^+) + \cos\theta^+(\tau_2^+) \\ &\quad + y^+(\tau_1^+)\sin(\tau_2^+ - \tau_1^+) - \cos(\theta^+(\tau_1^+) + \sigma_1(\tau_2^+ - \tau_1^+))], \\ &= [1 - \cos\tau_1^+ - (\beta + \tau_1^+)\sin\tau_1^+]\cos(\tau_2^+ - \tau_1^+) + 1 - \cos(\tau_2^+ - \tau_1^+) + \cos\theta_2 \\ &\quad + [\sin\tau_1^+ - (\beta + \tau_1^+)\cos\tau_1^+]\sin(\tau_2^+ - \tau_1^+) - \cos(\tau_1^+ + \tau_2^+ - \tau_1^+), \\ &= 1 - 2\cos\tau_2^+ - (\beta + \tau_1^+)\sin\tau_2^+ + \cos\theta_2, \\ &= 1 - 2\cos\tau_2^+ - (\beta + 2\tau_2^+ - 2\pi - \theta_2)\sin\tau_2^+ + \cos\theta_2.\end{aligned} \tag{46}$$

$$\begin{aligned}y^+(\tau_2^+) &= y^+(\tau_1^+)\cos(\tau_2^+ - \tau_1^+) + \sin(\tau_2^+ - \tau_1^+) \\ &\quad - \sigma_1[\sin\theta^+(\tau_2^+) + x^+(\tau_1^+)\sin(\tau_2^+ - \tau_1^+) - \sin(\theta^+(\tau_1^+) + \sigma_1(\tau_2^+ - \tau_1^+))], \\ &= [\sin\tau_1^+ - (\beta + \tau_1^+)\cos\tau_1^+]\cos(\tau_2^+ - \tau_1^+) + \sin(\tau_2^+ - \tau_1^+) - \sin\theta_2 \\ &\quad - [1 - \cos\tau_1^+ - (\beta + \tau_1^+)\sin\tau_1^+]\sin(\tau_2^+ - \tau_1^+) + \sin(\tau_1^+ + \tau_2^+ - \tau_1^+), \\ &= 2\sin\tau_2^+ - (\beta + \tau_1^+)\cos\tau_2^+ - \sin\theta_2, \\ &= 2\sin\tau_2^+ - (\beta + 2\tau_2^+ - 2\pi - \theta_2)\cos\tau_2^+ - \sin\theta_2.\end{aligned} \tag{47}$$

Because of the switch in the meanings of the inputs, it is now on the left trajectories that we must take account of the periodicity of θ when we substitute it for a time variable. We have therefore replaced θ_2 where it appears in a nonperiodic fashion with $2\pi + \theta_2$.

2.3.2 Crossover Trajectory R1

Using (40) we can replace (19), (20) and (21) with (remembering that $\sigma_1 = -1$)

$$\begin{aligned}\theta^-(\tau_1^-) &= 2\phi_0 + \sigma_1\tau_1^-, \\ &= -\tau_1^-.\end{aligned} \tag{48}$$

$$\begin{aligned}x^-(\tau_1^-) &= -\beta\sin\phi_0\cos\tau_1^- - \tau_1^-\sin\theta^-(\tau_1^-) + \sigma_1[1 - \beta\cos\phi_0\sin\tau_1^- - \cos\tau_1^-], \\ &= -\tau_1^-\sin(-\tau_1^-) - 1 + \beta\sin\tau_1^- + \cos\tau_1^-, \\ &= -1 + \cos\tau_1^- + (\beta + \tau_1^-)\sin\tau_1^-.\end{aligned} \tag{49}$$

$$\begin{aligned}y^-(\tau_1^-) &= -\beta\cos\phi_0\cos\tau_1^- - \tau_1^-\cos\theta^-(\tau_1^-) + \sin\tau_1^- + \sigma_1\beta\sin\phi_0\sin\tau_1^-, \\ &= -\beta\cos\tau_1^- - \tau_1^-\cos(-\tau_1^-) + \sin\tau_1^-, \\ &= \sin\tau_1^- - (\beta + \tau_1^-)\cos\tau_1^-.\end{aligned} \tag{50}$$

Then use (4), (48) and (8) to replace (22) and (25)

$$\begin{aligned}\theta^-(\tau_2^-) &= \theta^-(\tau_1^-) + 2\sigma_1(\tau_2^- - \tau_1^-), \\ &= \tau_1^- - 2\tau_2^-.\end{aligned} \tag{51}$$

$$\tau_1^- = 2\tau_2^- + \theta_2. \tag{52}$$

With (4), (49), (50) and (52) we can replace (28) and (29)

$$\begin{aligned}x^-(\tau_2^-) &= x^-(\tau_1^-)\cos(\tau_2^- - \tau_1^-) + \sigma_1[1 - \cos(\tau_2^- - \tau_1^-) + \cos\theta^-(\tau_2^-) \\ &\quad - \cos(\theta^-(\tau_1^-) + \sigma_1(\tau_2^- - \tau_1^-)) + y^-(\tau_1^-)\sin(\tau_2^- - \tau_1^-)], \\ &= [-1 + \cos\tau_1^- + (\beta + \tau_1^-)\sin\tau_1^-]\cos(\tau_2^- - \tau_1^-) - 1 \\ &\quad + \cos(\tau_2^- - \tau_1^-) - \cos\theta_2 + \cos(-\tau_1^- - \tau_2^- + \tau_1^-) \\ &\quad - [\sin\tau_1^- - (\beta + \tau_1^-)\cos\tau_1^-]\sin(\tau_2^- - \tau_1^-), \\ &= 2\cos\tau_2^- - 1 + (\beta + \tau_1^-)\sin\tau_2^- - \cos\theta_2, \\ &= 2\cos\tau_2^- - 1 + (\beta + 2\tau_2^- + \theta_2)\sin\tau_2^- - \cos\theta_2.\end{aligned} \tag{53}$$

$$\begin{aligned}y^-(\tau_2^-) &= y^-(\tau_1^-)\cos(\tau_2^- - \tau_1^-) + \sin(\tau_2^- - \tau_1^-) \\ &\quad - \sigma_1[\sin\theta^-(\tau_2^-) + x^-(\tau_1^-)\sin(\tau_2^- - \tau_1^-) - \sin(\theta^-(\tau_1^-) + \sigma_1(\tau_2^- - \tau_1^-))], \\ &= [\sin\tau_1^- - (\beta + \tau_1^-)\cos\tau_1^-]\cos(\tau_2^- - \tau_1^-) + \sin(\tau_2^- - \tau_1^-) + \sin\theta_2 \\ &\quad + [-1 + \cos\tau_1^- + (\beta + \tau_1^-)\sin\tau_1^-]\sin(\tau_2^- - \tau_1^-) - \sin(-\tau_1^- - \tau_2^- + \tau_1^-), \\ &= 2\sin\tau_2^- - (\beta + \tau_1^-)\cos\tau_2^- + \sin\theta_2, \\ &= 2\sin\tau_2^- - (\beta + 2\tau_2^- + \theta_2)\cos\tau_2^- + \sin\theta_2.\end{aligned} \tag{54}$$

2.3.3 Crossover Trajectory L2

Now it is trajectory L2 which consists of a single segment. The initial conditions become $\phi_0^+ \neq 0$ and (4) is followed for the entire interval $\tau \in [0, \tau_2^+]$. Using (4) and (39) with

$\sigma_1 = +1$ we get

$$\begin{aligned}\theta^+(\tau_2^+) &= 2\phi_0^+ + 2\sigma_1\tau_2^+, \\ &= 2\phi_0^+ + 2\tau_2^+.\end{aligned} \tag{55}$$

$$\phi_0^+ = \frac{2\pi + \theta_2 - 2\tau_2^+}{2}. \tag{56}$$

$$\begin{aligned}x^+(\tau_2^+) &= -\beta \sin\phi_0^+ \cos\tau_2^+ + \sigma_1[1 - \cos\tau_2^+ - \beta\cos\phi_0^+ \sin\tau_2^+ \\ &\quad + \cos\theta^+(\tau_2^+) - \cos(2\phi_0^+ + \sigma_1\tau_2^+)], \\ &= 1 - \cos\tau_2^+ - \beta\sin(\phi_0^+ + \tau_2^+) - \cos(2\phi_0^+ + \tau_2^+) + \cos\theta_2, \\ &= 1 - \cos\tau_2^+ - \beta\sin\tfrac{2\pi+\theta_2}{2} - \cos(\theta_2 - \tau_2^+) + \cos\theta_2.\end{aligned} \tag{57}$$

$$\begin{aligned}y^+(\tau_2^+) &= -\beta\cos\phi_0^+ \cos\tau_2^+ + \sin\tau_2^+ \\ &\quad - \sigma_1[-\beta\sin\phi_0^+ \sin\tau_2^+ + \sin\theta^+(\tau_2^+) - \sin(2\phi_0^+ + \sigma_1\tau_2^+)], \\ &= \sin\tau_2^+ - \beta\cos(\phi_0^+ + \tau_2^+) + \sin(2\phi_0^+ + \tau_2^+) - \sin\theta_2, \\ &= \sin\tau_2^+ - \beta\cos\tfrac{2\pi+\theta_2}{2} + \sin(\theta_2 - \tau_2^+) - \sin\theta_2.\end{aligned} \tag{58}$$

2.3.4 Crossover Trajectory R2

R2 now follows a two segment trajectory, like L2's from section 1.3.3 but with the modification that the first segment follows (40) instead of (6). Consequently, we use (48), (49) and (50) to find the coordinates at the end of the first segment of the crossover trajectory (at $\tau = \tau_1^-$), and then (5) and (8) with $\sigma_1 = -1$ to find

$$\theta^-(\tau_2^-) = \theta^-(\tau_1^-) = -\tau_1^-. \tag{59}$$

$$\tau_1^- = -\theta_2. \tag{60}$$

$$\begin{aligned}x^-(\tau_2^-) &= x^-(\tau_1^-)\cos(\tau_2^- - \tau_1^-) + \sigma_1[1 - \cos(\tau_2^- - \tau_1^-) - \cos\theta^-(\tau_1^-) \\ &\quad + y^-(\tau_1^-)\sin(\tau_2^- - \tau_1^-) + \cos(\theta^-(\tau_1^-) + \sigma_1(\tau_2^- - \tau_1^-))], \\ &= [-1 + \cos\tau_1^- + (\beta + \tau_1^-)\sin\tau_1^-]\cos(\tau_2^- - \tau_1^-) - 1 + \cos(\tau_2^- - \tau_1^-) + \cos\tau_1^- \\ &\quad - [\sin\tau_1^- - (\beta + \tau_1^-)\cos\tau_1^-]\sin(\tau_2^- - \tau_1^-) - \cos(-\tau_1^- - \tau_2^- + \tau_1^-), \\ &= -1 + \cos\tau_1^- + (\beta + \tau_1^-)\sin\tau_2^-, \\ &= -1 + \cos\theta_2 + (\beta - \theta_2)\sin\tau_2^-.\end{aligned} \tag{61}$$

$$\begin{aligned}y^-(\tau_2^-) &= y^-(\tau_1^-)\cos(\tau_2^- - \tau_1^-) + \sin(\tau_2^- - \tau_1^-) - \sigma_1[-\sin\theta^-(\tau_1^-) \\ &\quad + x^-(\tau_1^-)\sin(\tau_2^- - \tau_1^-) + \sin(\theta^-(\tau_1^-) + \sigma_1(\tau_2^- - \tau_1^-))], \\ &= [\sin\tau_1^- - (\beta + \tau_1^-)\cos\tau_1^-]\cos(\tau_2^- - \tau_1^-) + \sin(\tau_2^- - \tau_1^-) - \sin(-\tau_1^-) \\ &\quad + [-1 + \cos\tau_1^- + (\beta + \tau_1^-)\sin\tau_1^-]\sin(\tau_2^- - \tau_1^-) + \sin(-\tau_1^- - \tau_2^- + \tau_1^-), \\ &= \sin\tau_1^- - (\beta + \tau_1^-)\cos\tau_2^-, \\ &= -\sin\theta_2 - (\beta - \theta_2)\cos\tau_2^-.\end{aligned} \tag{62}$$

2.3.5 Crossover Computation

The procedure is the same as in section 1.3.5, except that the physically appropriate conditions (38) become

$$\begin{aligned} &\tau_1^- \leq \tau_2^-, \\ &\tau_1^+ \geq 0, \\ &\phi_0^+ \geq 0. \end{aligned} \tag{63}$$

The cases and their equations are

- For L1 with R1, equate (46) with (53) and (47) with (54) to solve for τ_2^+ and τ_2^-. Then (45) and (52) can be used to find τ_1^+ and τ_1^-.
- For L1 with R2, equate (46) with (61) and (47) with (62) to solve for τ_2^+ and τ_2^-. Then (45) and (60) can be used to find τ_1^+ and τ_1^-.
- For L2 with R1, equate (57) with (53) and (58) with (54) to solve for τ_2^+ and τ_2^-. Then (56) and (52) can be used to find ϕ_0^+ and τ_1^-.
- For L2 with R2, equate (57) with (61) and (58) with (62) to solve for τ_2^+ and τ_2^-. Then (56) and (60) can be used to find ϕ_0^+ and τ_1^-.

2.4 The Rest of the Barrier

The general procedure is the same as that outlined in section 1.4, with only a few differences in the equations used to take account of the switch of inputs.

2.4.1 Barrier Type L1

Subsets L1.1 and L1.2 are computed in the manner described in section 1.4.1 for the barrier type L1 with the pursuer at the origin, except for the modifications

- $\sigma_1 = +1$ for both L1.1 and L1.2
- $\zeta \in [0, \frac{2\pi+\theta_2}{2}]$ and $\eta = \frac{2\pi+\theta_2-2\zeta}{2}$ for L1.1
- $\eta_2 = \frac{\eta_1+2\pi+\theta_2}{2}$ for L1.2

2.4.2 Barrier Type R1

Subsets R1.1 and R1.2 are computed in the manner described in section 1.4.2 for the barrier type R1 with the pursuer at the origin, except for the modifications

- $\sigma_1 = -1$ for both R1.1 and R1.2
- $\zeta \in [\frac{\theta_2}{2}, 0]$ and $\eta = \frac{2\zeta-\theta_2}{2}$ for R1.1
- $\eta_2 = \frac{\eta_1-\theta_2}{2}$ for R1.2

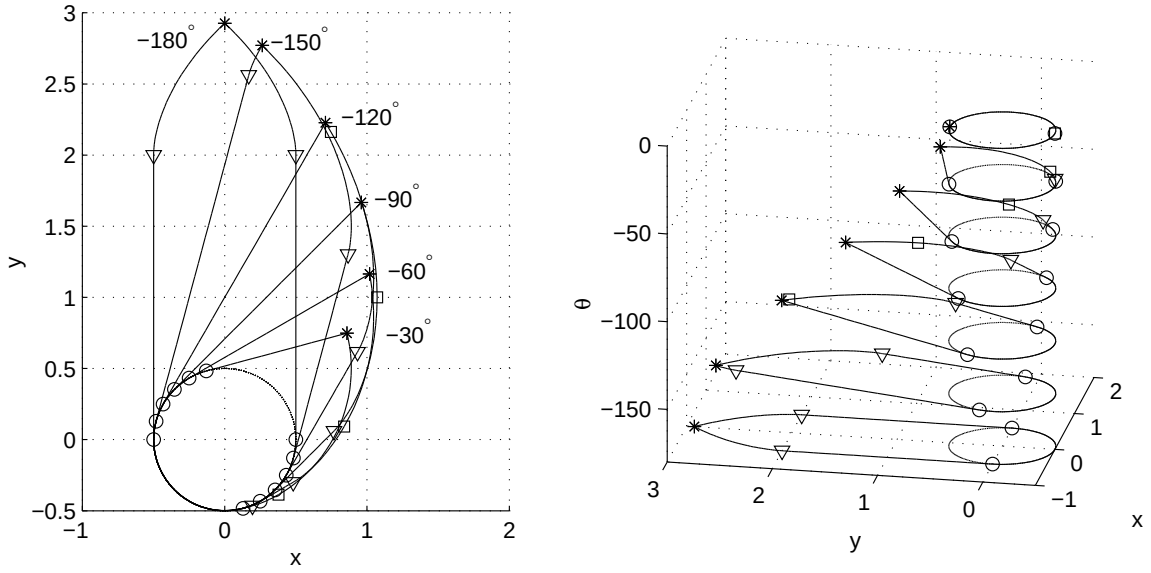

Figure 6: Cross sections of the barrier for $\beta = 0.5$ and the evader at the origin

2.4.3 Barrier Type L2

There is only one subset L2.1, which is computed in the same manner as L1.1 in section 2.4.1 except with a restricted range of $\zeta \in [\phi_0^+, \frac{2\pi+\theta_2}{2}]$.

2.4.4 Barrier Type R2

There are three subsets R2.1, R2.2, and R2.3; the first two are computed in the same manner as R1.1 and R1.2 in section 2.4.2. The third subset R2.3 is a single trajectory computed by by

1. Find $x(\tau_1^-)$ and $y(\tau_1^-)$ from (40) with x_0 and y_0 from (39) (with $\phi_0 = 0$) and $\sigma_1 = -1$.
2. Compute $x(\tau)$ and $y(\tau)$ from (5) for $\tau \in [\tau_1^-, \tau_2^-]$ with $x_0 = x(\tau_1^-)$, $y_0 = y(\tau_1^-)$, and $\sigma_1 = \sigma_2 = -1$.

R2.3 is this segment of trajectory $x(\tau)$ and $y(\tau)$ for $\tau \in [\tau_1^-, \tau_2^-]$.

2.5 Results

Figure 6 shows the same cross sections of the barrier as figure 5, but for the game with the evader at the origin. While the barrier at $\theta_2 = -180°$ is the same, the "points" of the other cross sections face into the capture region rather than out from it. As a result, the helical bulge of the three dimensional capture region for the game with the evader at the origin

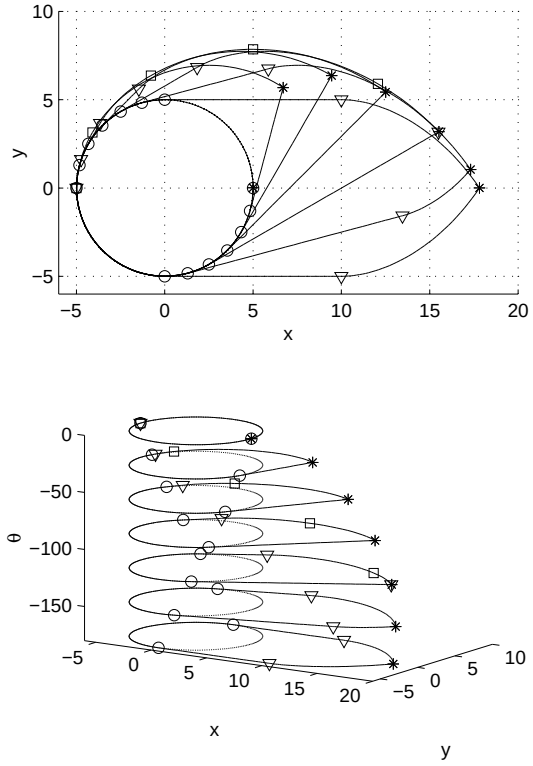

Figure 7: Cross sections of the barrier for the aircraft collision avoidance example

is narrower in the x direction but thicker in the θ direction than the corresponding helical bulge of the game with the pursuer at the origin.

The same trajectory combinations occur in the slices as occurred in figure 5: the upper two are L1 and R1, while the lower five are L2 and R2. The star, triangle, square and circle symbols mark the endpoints of the various barrier subsets.

Figure 7 shows cross sections of the barrier for the three dimensional aircraft collision avoidance example from [2, 3]. It is generated by computing the barrier for the $\beta = 1$ case, switching the x and y coordinates, and scaling those same coordinates by 5.

3 The Hamilton-Jacobi Formulation

We envision that a primary use of the computations above will be in the validation of numerical schemes for solving Hamilton-Jacobi (HJ) equations. This section discusses

- How to cast the determination of the barrier for the game of two identical vehicles into a HJ framework.
- How to use the analytic barriers determined in sections 1 and 2 to validate a numeric

solution to the appropriate HJ equation.

- The results obtained from our current numerical HJ solver and its convergence properties.

3.1 The Hamilton-Jacobi PDE

Because our own numeric schemes [3] have been based on level set methods, we may use some level set terminology in this section. In particular, we will encode surfaces like the barrier as the zero level set of some scalar function that we will call the level set function.

Let $\Psi(z, \tau)$ be our level set function (where z is the system state vector (2)). Ψ is the solution to the HJ equation

$$\frac{d\Psi}{d\tau} + H(z, \nabla\Psi) = 0. \tag{64}$$

Note that we run the HJ equation backwards in time from the terminal (capture) conditions. In order to solve this equation we must specify the HJ initial conditions $\Psi(z, 0)$, the boundary conditions, and the Hamiltonian $H(z, p)$ (where $p = \nabla\Psi$). The first two are the same regardless of whether the pursuer or the evader is at the origin, while the Hamiltonian differs slightly between the two games.

The initial conditions

$$\Psi(z, 0) = \sqrt{x^2 + y^2} - \beta \tag{65}$$

encode the capture cylinder in that $\{z : \Psi(z, 0) = 0\}$ is a cylinder of radius β running parallel to the θ axis. We have chosen initial conditions such that $\|\nabla\Psi(z, 0)\| = 1$, which is appropriate for our level set based schemes. However, since we are only interested in the zero level set of Ψ, any initial condition which produced the same zero level set would work—for example, $\Psi'(z, 0) = x^2 + y^2 - \beta^2$ might be better for other numeric HJ schemes. The boundary conditions are straightforward: periodic in θ and free in x and y.

The Hamiltonian with inputs is constructed from (1) (note that we multiply the dynamics by -1 to run backwards in time)

$$\begin{aligned} H(z, p, \sigma_1, \sigma_2) &= p_x(\sigma_1 y - \sin\theta) + p_y(1 - \sigma_1 x - \cos\theta) + p_\theta(\sigma_1 - \sigma_2), \\ &= p_y - p_y\cos\theta - p_x\sin\theta + \sigma_1(p_x y - p_y x + p_\theta) - \sigma_2 p_\theta, \end{aligned}$$

where we have denoted the first, second and third components of the gradient $p = \nabla\Psi$ as p_x, p_y and p_θ respectively. To get rid of the inputs we appeal to differential game theory [4], and at this point the two games' Hamiltonians diverge.

With the pursuer at the origin, σ_1 is the pursuer's input and seeks to maximize the size of the reachable region while σ_2 is the evader's input and seeks to minimize the size of the reachable region. Given (64), $|\sigma_1| \leq 1$ and $|\sigma_2| \leq 1$, it is not too hard to see that the optimal inputs are therefore

$$\begin{aligned} \sigma_1 &= -\operatorname{sign}(p_x y - p_y x + p_\theta), \\ \sigma_2 &= \operatorname{sign}(p_\theta). \end{aligned}$$

The resulting Hamiltonian for the pursuer at the origin is thus

$$H_P(z,p) = p_y - p_y\cos\theta - p_x\sin\theta - |p_x y - p_y x + p_\theta| + |p_\theta|. \tag{66}$$

The opposite is true with the evader at the origin: σ_1 is the evader's input and seeks to minimize the size of the reachable region while σ_2 seeks to maximize its size. The optimal inputs are therefore

$$\sigma_1 = \operatorname{sign}(p_x y - p_y x + p_\theta),$$
$$\sigma_2 = -\operatorname{sign}(p_\theta).$$

The resulting Hamiltonian for the evader at the origin is thus

$$H_E(z,p) = p_y - p_y\cos\theta - p_x\sin\theta + |p_x y - p_y x + p_\theta| - |p_\theta|. \tag{67}$$

Note that both of these Hamiltonians are continuous but not differentiable.

If we went ahead and solved (64) with H_P or H_E, then the zero level set of $\Psi(z,\tau)$ would contain only those states whose trajectories could reach the capture cylinder for the appropriate problem in exactly τ time units—it would omit those states whose trajectories either did not reach the cylinder or whose trajectories had passed completely through the cylinder after τ time units. If we are to compute the barrier, we must include any states whose trajectories could ever reach the capture cylinder, even if those trajectories could later exit the cylinder.

While such a computation can be accomplished in several ways, we describe the method with which we have had the most success. If the Hamiltonian in (64) is never negative, Ψ cannot increase as time advances, so the zero sublevel set of Ψ cannot shrink. Consequently, any state which passes into the reachable set (as given by the zero sublevel set) must always remain there. Therefore, instead of using the H_* (where $*$ is P or E) from (66) or (67) directly, use the modified Hamiltonian

$$\tilde{H}_*(z,p) = \max(0, H_*(z,p)).$$

After solving (64) with either $\tilde{H}_P$ or $\tilde{H}_E$, the barrier is given by the zero level set of $\Psi(z)$ where

$$\Psi(z) = \lim_{\tau\to\infty} \Psi(z,\tau).$$

3.2 Validation against the Analytic Barrier

If $\Psi(z)$ has been determined via some numerical scheme for solving the HJ equation and $\|\nabla\Psi(z)\| = 1$, the following technique can be used to validate its results against the appropriate analytically determined barrier from either section 1 or 2. Given a point z^* on the analytic barrier, the error in the numeric solution to the HJ equation for that point is $|\Psi(z^*)|$—if z^* is not a grid point, use a sufficiently high order interpolation to compute $\Psi(z^*)$; for example, Matlab's `interp3` command. Given a large number of z^*, statistics on maximum and average error can be computed. It is important that the crossover points are included among the z^*, since it is at these nonsmooth points on the barrier that the maximum error in the HJ solution is likely to occur.

3.3 Our Current Results

We conclude this section with some comments on our own experiences in solving the HJ formulation of these games. We have developed an HJ solver on an equally spaced Cartesian grid in C++ using 5^{th} order accurate Weighted Essentially Non-Oscillatory (WENO) stencils for the gradient approximation, a Lax-Friedrichs numerical approximation for the Hamiltonian, and a 2^{nd} order accurate Total Variation Diminishing (TVD) Runge-Kutta time integrator. We solve the HJ PDE only near the zero level set using an original version of what has variously been called "narrowbanding" or "local level sets". Stability near the boundaries is preserved partly through repeated reinitialization to keep $\|\Psi\| = 1$—for more details and references, see [3].

We have found in executions of our implementation that the bulge for the game with the pursuer at the origin is so razor thin at its edge that our attempts to compute the HJ solution have been unable to come anywhere near resolving the full extent of the bulge.

In contrast, the version of the game with the evader at the origin has a bulge thick enough that we have been able to resolve it well even on fairly coarse grids. Figure 8 shows the zero level set of $\Psi(z)$ computed by our numerical scheme on a 100^3 grid as a solid surface. Overlaid on top of this surface are 2600 points from the barrier, as computed by the Matlab function `plotFig7.m` described in section 4.3.

Using the script `errorLS.m` described in section 4.3, we performed a convergence analysis of the error in our computed $\Psi(z)$ versus approximately 60000 points on the barrier; the results are shown in figure 9. The solution appears to be convergent in both maximum pointwise and average error. In addition, the maximum pointwise error is never significantly more than a grid cell width. We have not yet determined why a (5,2) scheme is showing only a linear convergence rate—possible culprits include the loss of high order accuracy near the derivative discontinuities in the solution, or possibly the interpolation scheme used to compute the errors in Matlab.

4 Matlab

The code is available from the author [5]. There are three sets of files (divided by directory).

4.1 Pursuer at the Origin

The directory `PursuerOrigin` contains the files necessary to recreate the game that Merz describes, following the algorithm in section 1. Some additional files produce the figures for this report.

`barrier.m`: Generates a collection of points on the barrier for a particular choice of θ_2 and β. Left, right and capture circle portions of the barrier are returned separately, as are points on the left and right barrier separating the barrier subsets; for example, the "curved" point separates L1.2 and L1.1 on an L1 barrier. This function basically

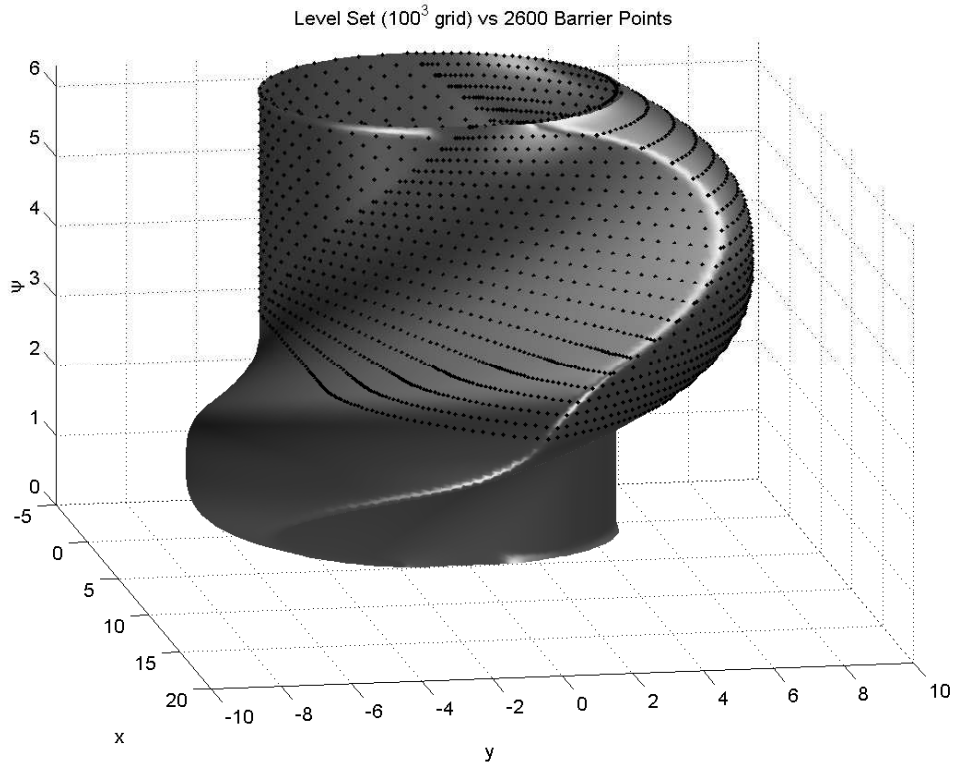

Figure 8: Barrier (evader at origin) as computed by solving (64) with Hamiltonian $\tilde{H}_E$

implements the procedures from section 1.4.

`crossover.m`: Computes the crossover point for a particular choice of θ_2 and β. Returns details of the crossover trajectory types, the parameters $\tau_1^{\pm}$, $\tau_2^{\pm}$, ϕ_0^{-} and the crossover point. Implements the procedures from section 1.3.

`optTraj.m`: Helper function for `barrier`. Computes points on an optimal trajectory. Implements the equations (4), (5) and (6).

`optTraj2.m`: Helper function for `barrier`. Computes points on a two segment optimal trajectory, as needed by barrier types L1, R1 and L2.

`equality.m`: Helper function for the `fsolve` in `crossover`. Equates the x and y coordinates of the crossover trajectories.

`crossX.m`: Helper function for `equality`. Implements the crossover trajectories in the x coordinate: (17), (32), (28) and (36).

`crossY.m`: Helper function for `equality`. Implements the crossover trajectories in the y coordinate: (18), (33), (29) and (37).

`trajODE.m`: Implements the system dynamics (1) for use with `ode23`.

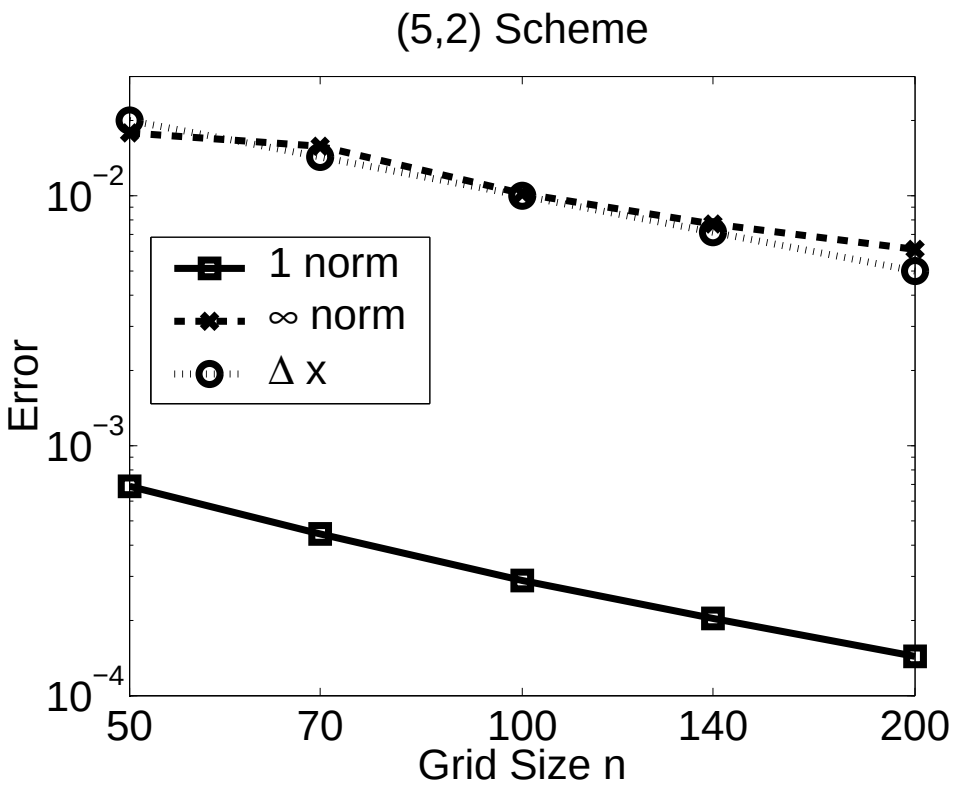

Figure 9: Error in $\Psi(z)$ when compared to analytically computed barrier

`plotFig7.m`: Recreates Merz's Figure 7 using `barrier`. Used to generate figure 5.

`plotBarrier.m`: Used to generate figure 2.

`plotCrossover.m`: Used to generate figure 3.

`plotLeftBarrier.m`: Used to generate figure 4.

4.2 Evader at the Origin

The directory `EvaderOrigin` contains the files necessary to recreate the game with the evader at the origin, following the algorithm in section 2. The functions and names of the files are the same as those described in section 4.1, with the appropriate modifications to place the evader at the origin.

4.3 The Analytic Solution of the Hamilton-Jacobi Problem

The directory `Converge` contains files which are set up to test whether our implementation of a numerical Hamilton-Jacobi solver is convergent to the true solution as determined above. Because of the difficulty described at the end of section 3 in numerically solving

the Hamilton-Jacobi problem with the pursuer at the origin, we have only implemented convergence tests for the version of the game with the evader at the origin. Consequently, the functions `barrier`, `crossover`, `optTraj`, `optTraj2`, `equality`, `crossX` and `crossY` in this directory are the same as those from the directory `EvaderOrigin` (see section 4.2).

`plotFig7.m`: This file is now a function which generates a large number of points on a collection of barrier slices. The exact number of points is controlled by the variable `dt` set in `barrier` and the number of slices set in `plotFig7`. The choice of `dt` $= 0.01$ and an interslice spacing of 0.01π produce about 60000 points. This function also scales, shifts and swaps the coordinates to agree with the aircraft example from [2, 3].

`errorLS.m`: Given a level set $\Psi(z)$ and a collection of points on the barrier from `plotFig7`, this function computes the pointwise error of $\Psi(z)$ in several different norms using interpolation. Currently, the level set is loaded from a file by the `loadfunc` call; the user should replace this call with his or her own method for getting the level set. Note that this code was written to demonstrate mere convergence (ie first order accuracy), and thus linear interpolation was sufficient.

To make an error analysis of a Hamilton-Jacobi solver:

- Implement the Hamilton-Jacobi solver for (64) with Hamiltonian (67), initial conditions (65) (with $\beta = 1$), periodic boundary conditions in θ, and free boundary conditions in x and y. Figure 7 indicates suitable bounds for the computational grid (remember that $\theta \in [-\pi, \pi]$ or some equivalent domain, even though the figures show only the bottom half of this domain).
- Evolve $\Psi(z, \tau)$ according to (64). Around $\tau = 2.6$, $\Psi(z, \tau)$ will have reached a steady state $\Psi(z)$.
- If necessary, update $\Psi(z)$ so that $\|\nabla\Psi(z)\| = 1$ using a sufficiently high order reinitialization procedure (for references on reinitialization, see [3]).
- In Matlab, use `plotFig7` to generate a collection of barrier points. Scale the x and y coordinates by five and swap them. If necessary, shift the θ coordinate; for example, we used $\theta \in [0, 2\pi]$ for our level set computation, so we had to add 2π to the θ coordinates from `plotFig7`.
- Modify `errorLS` to load $\Psi(z)$ and compute pointwise error estimates.

5 Conclusions

We have managed to recreate Merz's analysis for the game of two identical cars with the pursuer at the origin, and to extend it to the game with the evader at the origin. As a result, we can analytically derive points that lie on the zero level set of the solution to a nonlinear, continuous but not differentiable Hamilton-Jacobi partial differential equation.

These points can be used to determine the convergence properties of three dimensional numerical Hamilton-Jacobi solvers.

Acknowledgments: We would like to thank Rodney Teo for bringing Merz's work to our attention, and Professor Ron Fedkiw for suggesting the method of validating numerical Hamilton-Jacobi solutions against the analytic barrier computation.

References

[1] A. W. Merz, "The Game of Two Identical Cars", *Journal of Optimization Theory and Applications*, vol. 9, no. 5, pp. 324–343 (1972).

[2] C. J. Tomlin, *Hybrid Control of Air Traffic Management Systems.* Ph.D. thesis, Department of Electrical Engineering, University of California, Berkeley, 1998.

[3] I. Mitchell, A. M. Bayen, and C. J. Tomlin, "Validating a Hamilton-Jacobi Approximation to Hybrid System Reachable Sets", in *Hybrid Systems: Computation and Control* (M. D. DiBenedetto and A. Sangiovanni-Vincentelli, eds.), LNCS 2034, pp. 418–431, Springer Verlag, 2001.

[4] T. Başar and G. J. Olsder, *Dynamic Noncooperative Game Theory* 2^{nd} ed., Philadelphia: SIAM, 1999.

[5] At the time of writing, `http://www-sccm.stanford.edu/~mitchell`