Chapter 3
Viability Problems in Robotics

3.1 Introduction

This chapter studies three applications to robotics, one focussing on field experiments of the viability feedback allowing a robot to rally a target in an urban environment while avoiding obstacles, the second one dealing with the safety envelope of the landing of a plane as well as the regulation law governing the safe landing evolutions viable in this envelope, and the third one focused on navigation of submarines in rivers.

3.2 Fields Experiment with the Pioneer

Viability theory not only computes the capture basin of the target viable in a given environment, but also provides the dedicated feedback piloting the state variables towards the target from any state of the capture basin. For this purpose, the time is discretized in time steps, the environment and the target are stored in a grid, and the viability software uses the capture basin algorithm to compute the graph of the feedback represented as a subset of a grid of state-control pairs. The graph of this feedback is integrated in the embedded software of the robot. At each time step, sensors locate the position of the robot and its direction. The feedback provides the controls.

The experimental robot presented is a Pioneer 3AT of *activmedia robotics* and the environment is a road network, on which a target to be reached in minimal time has been assigned.

Two sensors (odometers and GPS) are used for its localization, but the robot does not use the other sensors locating the obstacles and the target.

J.-P. Aubin et al., *Viability Theory*, DOI 10.1007/978-3-642-16684-6_3,

Fig. 3.1 Pioneer Robot used for the field experiment.
View of the Pionner, its "localization" sensors (GPS, left *and odometer,* right*) and the computer containing the tabulated graph of the viable feedback map (see Fig. 3.2, p.107) connected to the navigation system of the robot.*

The variables, controls, dynamics, constraints and target are as follow:

1. **State variables:** x_i, $i = 1, 2$, positions of the vehicle, θ, heading (direction),
2. **Controls:** u, total velocity, ω, angular velocity
3. **Control System**

$$\begin{cases} (i) & x_1'(t) = u(t)\cos(\theta(t)) \\ (ii) & x_2'(t) = u(t)\sin(\theta(t)) \\ (iii) & \theta'(t) = \omega(t) \end{cases} \tag{3.1}$$

4. **State Constraints and Target:**
   The position constraints and targets have been discretized on a grid (see the left figure of Fig. 1.11, p.23) and the direction is bounded: $\theta \in [\theta^\flat, \theta^\sharp]$.
5. **Constraints on controls:** Constraints are added which encode position dependent bounds on velocity and angular rotation velocity:

$$u \in [-\mu^\flat(x_1, x_2, \theta), \mu^\sharp(x_1, x_2, \theta)], \ \& \ \omega \in [-\omega^\flat(x_1, x_2, \theta), \omega^\sharp(x_1, x_2, \theta)]$$

Knowing the graph of feedback map computed by the viability algorithm, which is a five-dimensional set of vectors (x, y, θ, u_1, u_2), at each time step,

- The sensors measure the position and the heading (x, y, θ);
- The software takes into account this state of the robot and provides the controls (u_1, u_2) such that (x, y, θ, u_1, u_2) belongs to the graph of the feedback;
- These controls are transmitted to the navigation software of the Pioneer robot which waits for the next time step to update the velocity controls for moving to the next position

and repeats these steps.

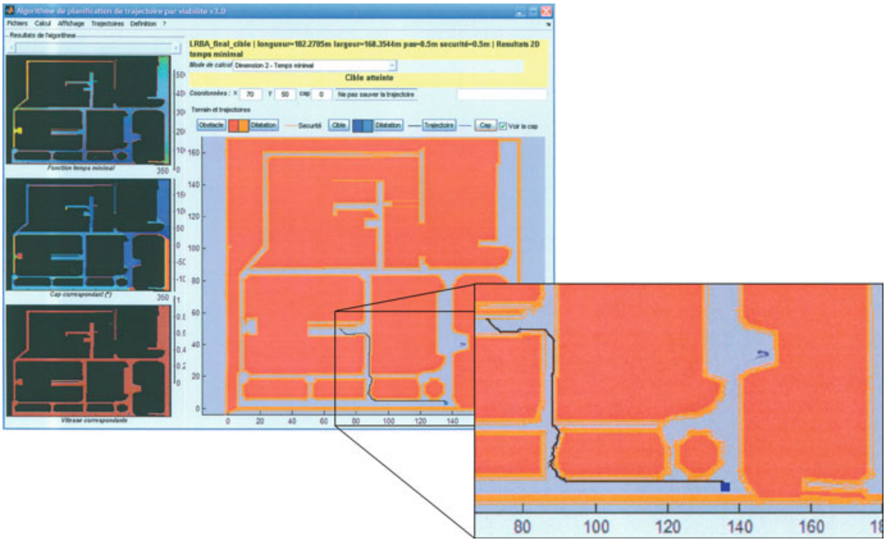
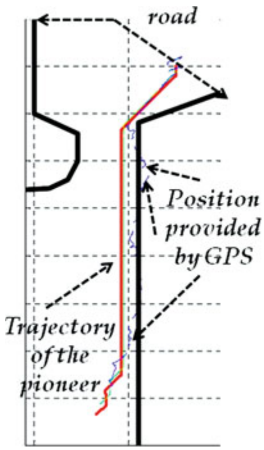


Fig. 3.2 Computation of the feedback for the Pioneer Robot.
The left column of the computer screen displays three windows providing in color scales the values of the minimal time at each position in the road (first window from the top), the direction (second window) and the velocities (provided by the feedback map). The main window displays the map of the road network and the trajectory of the robot, which is enlarged in the second figure.

This viability software was used for field implementation.



The robot must remain on the road, represented by thick lines. The actual trajectory of the robot rallying the target in minimum time while remaining in the road is indicated in grey. This actual trajectory is subjected to position errors estimated from the odometer and the GPS (providing sometimes positions outside the road). The dedicated feedback computed by the viability algorithm taking into account the environment allows the software to correct these errors.

3.3 Safety Envelopes in Aeronautics

In this section, we present an application of viability theory to *safety analysis in aeronautics*. We focus on the landing portion of the flight of a large civilian airliner (DC9-30).

One of the key technologies for design and analysis of safety critical and human-in-the-loop systems is *verification*, which allows for heightened confidence that the system will perform as desired. Verification means that from an initial set of states (for example, aircraft configurations, or states, such as position, velocity, flight path angle and angle of attack), a system can reach another desired set of states (*target*) while remaining in an acceptable range of states, called *envelope* in aeronautics, (i.e., an environment, in the viability terminology). The subset of states from which the target can be reached while remaining in the envelope is sometimes called the set of *controllable states* or *maximal controllable set* in aeronautics: in viability terminology, this is the *capture basin* of the target viable in the envelope. For example, if an aircraft is landing, the initial set of states is the set of acceptable (viable) aircraft configurations of the aircraft a few hundred feet before landing, the target is the set of acceptable aircraft states at touch down, and the envelope is the range of states in which it is safe to operate the aircraft. A *safe landing evolution* is one which starts from the envelope until it reaches the target in finite time.

Viability theory provides a framework for computing a capture basin of a given target, and thus, the maximal controllable set.

The benefit of this approach, is that it provides a *verification* (for the mathematical models used) that the system will remain inside the envelope and reach target.

This is an "inverse approach" (see Box 2, p. 5), to be contrasted with "direct approaches" (see Box 1, p. 5), using simulation methods, such as Monte-Carlo methods. They do not provide any guarantee that evolutions starting from configurations that are not part of the testing set of the simulation will land safely. Monte Carlo methods have historically been used to explore the possible evolutions a system might follow. The more finely gridded the state-space, the more information the Monte Carlo simulations will provide. However, this class of methods is fundamentally limited in that it provides no information about initial conditions outside the grid points. This provides a good motivation for the use of viability techniques: they provide both capture basin (maximal controllable set) and the a posteriori feedback governing safe evolutions, i.e., the *certificate guaranteeing* that from this *set*, an evolution will land safely if piloted using this feedback. Monte-Carlo methods may provide such initial states and evolutions piloted with a priori feedbacks if one is... lucky.

The first subsection of this section presents the model of the longitudinal dynamics of the aircraft, as well as the definition of the safety envelopes in the descent (flare) mode of the aircraft. The following subsection presents the computation of both the capture basin of the touch down target and the a posteriori *viability feedback* dedicated to safe touch down, in the sense that is its not given a priori, but computed from the target and the environment.

3.3.1 Landing Aircraft: Application of the Capture Basin Algorithm

3.3.1.1 Physical Model, Equations of Motion

This section presents the equations of motion used to model the aircraft's behavior.

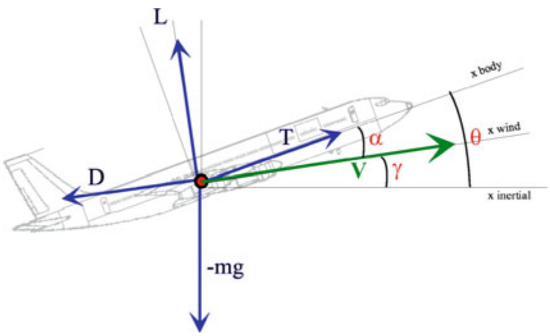


Fig. 3.3 Aircraft dynamic model.
Point mass force diagram for the longitudinal dynamics of the aircraft. Equation (3.2) is given in the inertial frame. V, γ, α, L, D *and* T *are used in (3.2) as displayed here.*

The aerodynamic properties of the aircraft, which are used for this model are derived from empirical data as well as fundamental principles. We model the longitudinal dynamics of an aircraft (see Fig. 3.3).

15 States and Controls. The state variables are:

1. the velocity V,
2. the flight path angle γ,
3. The altitude z.

We denote $x = (V, \gamma, z)$ the state of the system.
The controls (inputs) are:

1. the thrust T
2. the angle of attack α.

(to be controlled by the pilot or the autopilot). We denote $u = (T, \alpha)$ the control of the system.

We consider a point mass model where the aircraft is subjected to the gravity force mg, thrust T, lift L and drag D. We call m the mass of the aircraft. The equations of motion for this system read:

$$\frac{d}{dt}\begin{bmatrix} V \\ \gamma \\ z \end{bmatrix} = \begin{bmatrix} \frac{1}{m}[T\cos\alpha - D(\alpha, V) - mg\sin\gamma] \\ \frac{1}{mV}[T\sin\alpha + L(\alpha, V) - mg\cos\gamma] \\ V\sin\gamma \end{bmatrix} \tag{3.2}$$

Typically, landing is operated at $T_{\text{idle}} = 0.2 \cdot T_{\max}$ where $T_{\max}$ is the maximal thrust. This value enables the aircraft to counteract the drag due to flaps, slats and landing gear. Most of the parameters for the DC9-30 can be found in the literature. The values of the numerical parameters used for the DC9-30 in this flight configuration are $m = 60,000\,\text{kg}$, $T_{\max} = 160,000\text{N}$ and $g = 9.8\,\text{ms}^{-2}$. The lift and drag forces are thus

$$\begin{array}{rcl} L(\alpha, V) & = & 68.6\ (1.25 + 4.2\alpha)V^2 \\ D(\alpha, V) & = & [2.7 + 3.08\ (1.25 + 4.2\alpha)^2]V^2 \end{array} \tag{3.3}$$

(they are expressed in Newtons if V is taken in m/s). We refer to the specialized literature for the methods leading to these formulas.

3.3.1.2 Landing Maneuvers

In a typical autoland maneuver, the aircraft begins its approach approximately 10 nautical miles from the touchdown point. The aircraft descends towards the *glideslope*, an inertial beam which the aircraft can track. The landing gear is down, and the pilot sets the flaps at the first high-lift configuration in the landing sequence.

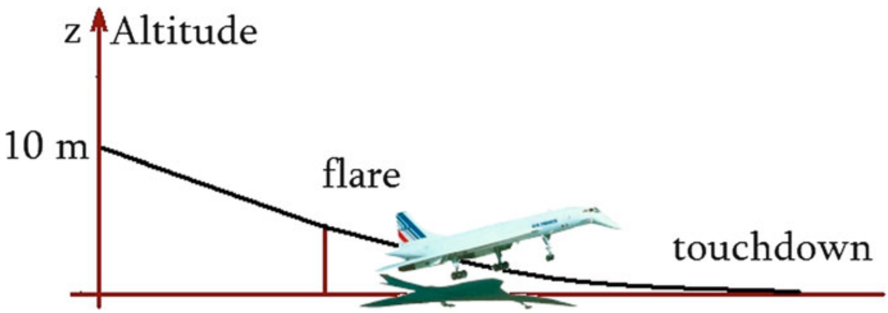


Fig. 3.4 Autoland Profile.
Typical Autoland Profile and flare section of the flight before touch down studied in this example.

The autopilot captures the glideslope signal around five nautical miles from the touch down point. The pilot increases flap deflection to effect a descent without increasing velocity. The pilot steps the flaps through the different flap settings, reaching the highest deflection when the aircraft reaches 1,000 feet altitude. At approximately 50 feet, the aircraft leaves the glideslope and begins the flare maneuver, which allows the aircraft to touchdown smoothly on the runway with an appropriate descent rate. The deflection of the slats is correlated with the deflection of the flaps in an automated way.

3.3.1.3 Safety Envelopes

Flight operating conditions are defined by the limits of aircraft performance, as well as by airport or the Federal Aviation Administration (FAA) regulations. During descent and flare, the aircraft proceeds through successive flap and slat settings. In each of these settings, the safe set is defined by bounds on the state variables. The maximal allowed velocity $V_{\max}$ is dictated by regulations. The minimal velocity is related to the stall velocity by $V_{\min} = 1.3 \cdot V_{\text{stall}}$. The minimal velocity is an FAA safety recommendation, the aircraft might become uncontrollable below V_{stall}.

During descent, the aircraft tracks the glideslope and must remain within $\pm 0.7°$ of the glideslope angle γ_{GS}. By regulation, as the aircraft reduces its descent rate to land smoothly (in the last 50 feet before touch down), the flight path angle γ in "flare mode" can range from $\gamma_{\min} = -3.7°$ to 0.

During descent and flare, thrust T should be at idle, but the pilot can use the full range of angle of attack α. Other landing procedures in which T varies are used as well.

16 Flight and Touch down envelopes: The environment and the target. The environment

$$K_{\text{flare}} := [V_{\min}, V_{\max}] \times [\gamma_{\min}, 0] \times \mathbb{R}_+ \tag{3.4}$$

is the flight envelope in flare mode, or *flare envelope*, when the velocity lies between minimal velocity $V_{\min} = 1.3 \cdot V_{\text{stall}}$ and a maximal velocity, when the negative path angle is larger than $\gamma_{\min} = -3.7°$ and the altitude is positive.
The target

$$\begin{cases} C_{\text{touch down}} := \\ \{(V, \gamma, z) \in [V_{\min}, V_{\max}] \times [\gamma_{\min}, 0] \times \{0\} \text{ such that } V \sin\gamma \geq \dot{z}_0\} \end{cases} \tag{3.5}$$

is the *touch down envelope*, when the velocity and the negative path angle are constrained by $V \sin\gamma \geq \dot{z}_0$ and the altitude is equal to 0.
The constraints on the controls are $U(x) := [T_{\min}, T_{\max}] \times [\alpha_{\min}, \alpha_{\max}]$

At touch down ($z = 0$, but not $V = 0$ which describes the stopping time of the aircraft), the restrictions are the same as flight parameters of the flight envelope, except for the descent velocity. This last requirement becomes $\dot{z}(t) > \dot{z}_0$, where $\dot{z}_0$ represent the maximal touch down velocity (in order not to damage the landing gear). This condition thus reads $V \sin\gamma \geq \dot{z}_0$.

The physical constraints of this problem are thus defined in terms of the safety envelope and the target, mathematically expressed by (3.4) and (3.5) respectively.

We now illustrate the difficulty of keeping the system inside the constraint set K_{flare} defined by (3.4) and of bringing it to the target $C_{\text{touch down}}$ defined by (3.5).

The environment (the flight envelope in flare mode) K_{flare} is a box in the (V, γ, z) space. Similarly, the target $C_{\text{touch down}}$ is a subset of the $z = 0$ face of K_{flare}, defined by $0 \geq V \sin\gamma \geq \dot{z}_0$ and $V \in [V_{\min}, V_{\max}]$.

Note that it is common for landing maneuvers to operate at constant thrust, which does not mean that, in the current model, the controls are constant, it only means that one of the control variables is constant for the duration of the maneuver (or some portion of it).

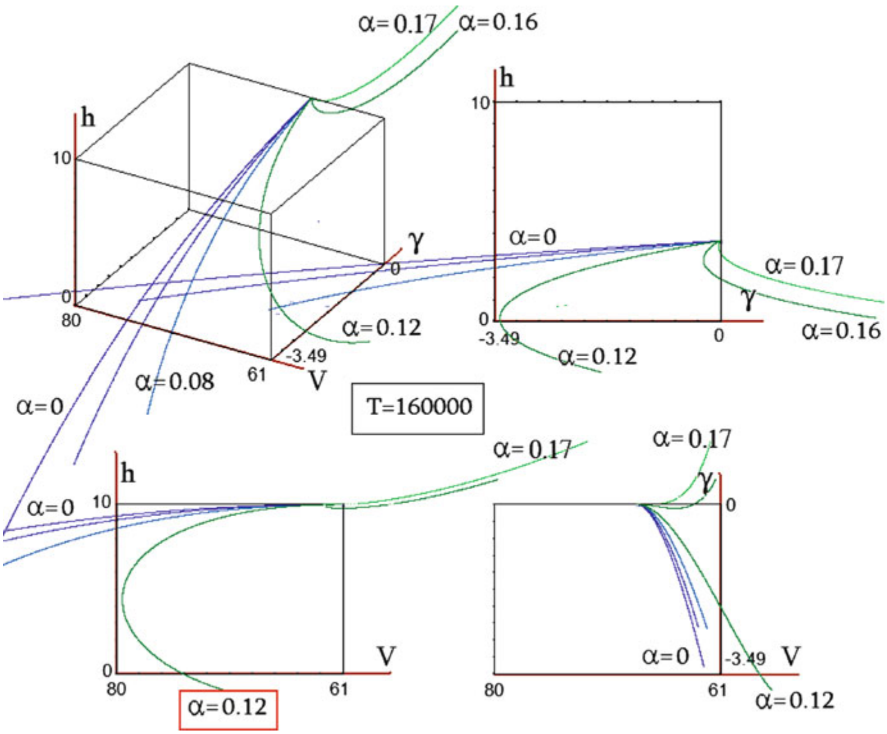


Fig. 3.5 Trajectories of evolutions with constant thrust.
Example of trajectories of evolutions emanating from a point at the $\gamma = 0$ face of K_{flare}, with maximal thrust and different angles of attack α. Some of these evolutions are not viable in K_{flare} and do not reach C in finite time.

Figure 3.5 displays several trajectories of evolutions obtained by applying $T = T_{\max}$ with different values of α. As can be seen from this figure, the trajectories obtained in the (V, γ, z) space exit K_{flare} in finite time. This means that the control $(\alpha, T_{\max})$ does not enable one to reach the target C starting from the corresponding initial condition.

Similarly, Fig. 3.6 shows trajectories of several evolutions in the (V, γ, z) space obtained with a fixed α and different values for T. As can be seen from this figure, some of these trajectories are not viable in K and do not reach the target $C_{\text{touch down}}$ before they hit the ground of exit K_{flare}.

This can be seen for the $T = 0$ and $T = 20,000$ trajectories, which start from the point B in Fig. 3.6, and exit K at B_1. Note that other trajectories stay in K, for example the one obtained for $T = 160,000$, which manages to reach the ground safely. Maneuvers can include several sequences for which some of the parameters are fixed, in the present case, using a constant thrust for descent is a common practice in aviation. The figure enables one to see the evolution of the flight parameters, V and z as the altitude decreases (see in particular top right subfigure).

This figure also displays the trajectory of one viable evolution emanating from B which reaches the target C in finite time. This evolution is governed by the a posteriori viability feedback computed by the Capture Basin Algorithm providing minimal time evolutions.

These two figures illustrate the difficulty of designing viable evolutions for this system, which thus underlines the usefulness of the Capture Basin Algorithm.

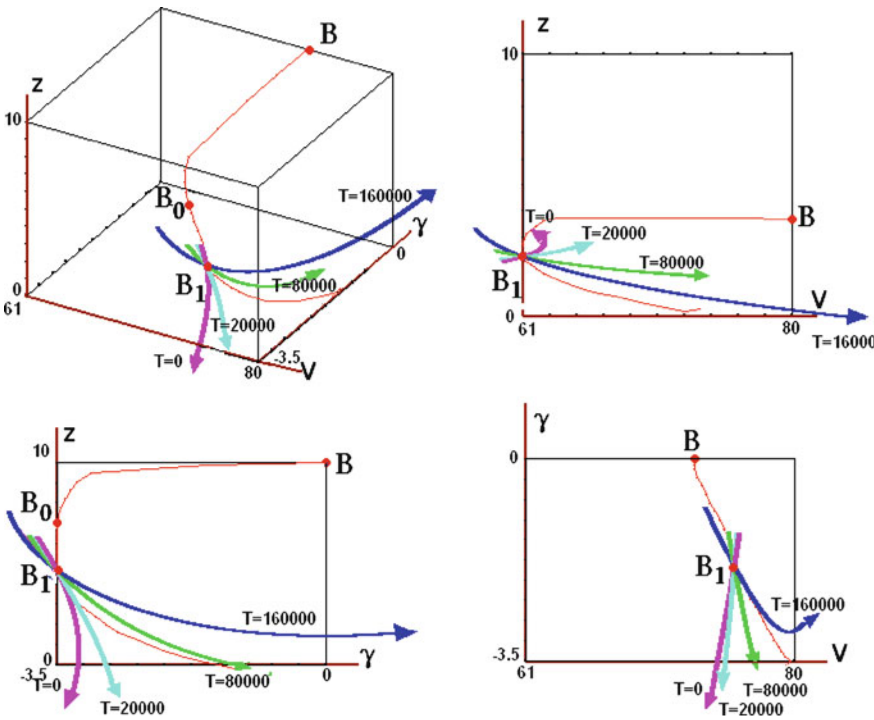


Fig. 3.6 Trajectories of evolutions with constant thrust.
Example of trajectories of evolutions emanating from a point at the $\gamma = 0$ face of K_{flare}, with different thrusts and a fixed angle of attack α. Some of these trajectories are not viable in K_{flare} and do not reach $C_{\text{touch down}}$ in finite time. However, the trajectory of the evolution starting at B and governed by the viable feedback synthesized by the Capture Basin Algorithm is viable.

3.3.2 *Computation of the Safe Flight Envelope and the Viability Feedback*

In the process of landing described in the previous sections, the following question is now of interest: starting from a given position in space (altitude z), with given flight conditions (velocity V and flight path angle γ), with fixed thrust, is there a viable feedback map (i.e., a switching policy made of a set of successive flap deflections/retractions), for which there exists a control (angle of attack α) able to bring the aircraft safely to the ground?

The *safe touch down envelope* is the set of states (V, γ, z) where $z > 0$ (aircraft in the air), from which it is possible to touch down the ground safely, (i.e. to reach $C_{\text{touch down}}$) while doing a safe flare (i.e. while staying in K_{flare}). It is therefore the capture basin

$$\mathrm{Capt}_{(3.2)}(K_{\text{flare}}, C_{\text{touch down}}) \tag{3.6}$$

of the target viable in K_{flare}.

We now provide both the capture basin and the graph of the a posteriori viability feedback computed in terms of the target and the environment by the *Capture Basin Algorithm*. Knowing this a posteriori viability feedback dedicated to this task, one knows from the theorems and still can "*verify*" that any evolution starting from the capture basin (the safe flight envelope) and governed by this dedicated a posteriori feedback is viable until it reaches the target.

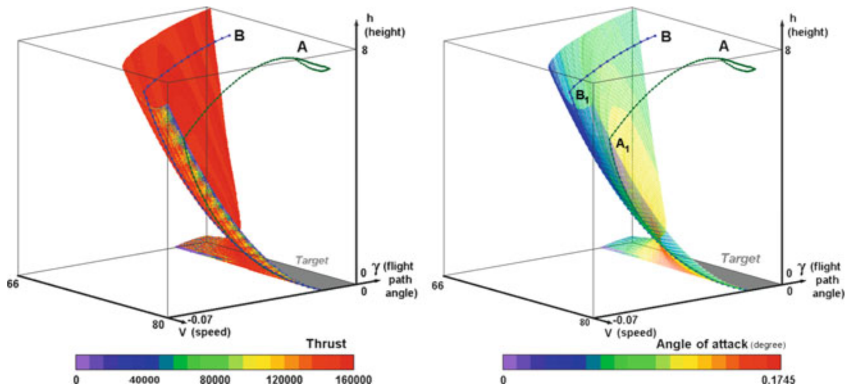


Fig. 3.7 Capture basin, thrust and angle of attack feedback.
Left: The thrust component of the feedback to apply at the boundary of $\mathrm{Capt}_{(3.2)}(K_{\mathit{flare}}, C_{\mathit{touch\ down}})$ *to maintain the aircraft in the safe flight envelope. The thrust is indicated by the colorimetric scale on the boundary of the capture basin* $\mathrm{Capt}_{(3.2)}(K_{\mathit{flare}}, C_{\mathit{touch\ down}})$. *Right: The angle of attack component is samely indicated by the colorimetric scale on the boundary of the capture basin with the trajectories of two viable evolutions of the landing aircraft.*

Figure 3.7 (left and right) shows the west boundary of $\mathrm{Capt}_{(3.2)}(K_{\mathrm{flare}}, C_{\mathrm{touch\ down}})$, Fig. 3.7 (left) the thrust component and Fig. 3.7 (right) the angle of attack of the feedback map $(V, \gamma, z) \mapsto (T, \alpha) \in R(V, \gamma, z)$ needed on the boundary of $\mathrm{Capt}_{(3.2)}(K_{\mathrm{flare}}, C_{\mathrm{touch\ down}})$ to govern a safe evolution of the aircraft in K_{flare} until landing at $C_{\mathrm{touch\ down}}$.

The target $C_{\mathrm{touch\ down}}$ is not represented explicitly in Fig. 3.7, but one can visually infer it from

$$C_{\mathrm{touch\ down}} \;=\; \mathrm{Capt}_{(3.2)}(K_{\mathrm{flare}}, C_{\mathrm{touch\ down}}) \cap ([V_{\min}, V_{\max}] \times [\gamma_{\min}, 0] \times \{0\})$$

It is the set of values of (V, γ) such that $\dot{z}_0 \leq V \sin\gamma \leq 0$, which is the almost rectangular slab at the top of the grey bottom of the cube. Reaching any (V, γ) value in this slab represents a safe touch down. Several trajectories are represented in this figure, one starting from the point A, evolving inside the capture basin until it reaches the boundary, at point A_1. At that point, it has to follow the feedback prescribed by the capture basin algorithm, to stay inside K, until it lands safely at the end of the trajectory. The evolution between B and B_1 is of the same type, it reaches the boundary at B_1, and then needs to apply the feedback to reach the target safely.

As can be seen, the thrust needs to be maximal almost everywhere far from the ground, which is intuitive: close to $\gamma = \gamma_{\min}$, high thrust is needed to prevent the aircraft from descending along a too steep flight path angle, or reaching $z = 0$ with a too fast descent velocity. Note that close to the $V = V_{\max}$ boundary of K_{flare}, lower values of T are required, which again is intuitive: in order not to exceed the maximal velocity required by K_{flare}, a low value of thrust must be applied. Finally, close to the ground (low values of z and V), low thrust is required to touch down, since the aircraft is almost on the ground.

The evolutions of the flight parameters $(V(t), \gamma(t), z(t))$, as well as the evolution of the viable feedback $(T(t), \alpha(t))$ are displayed in Figs. 3.8 and 3.9 for the two evolutions emanating from B and A respectively.

1. **Trajectory emanating from** A.

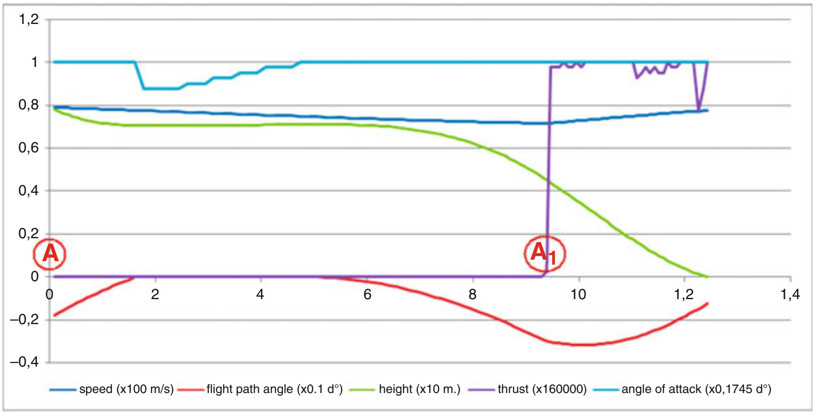


Fig. 3.8 Graph of the evolution starting from A.
Evolutions of the flight parameters and the control variables for the landing starting from state A.

We first describe the trajectory emanating from the point A. This corresponds to a descent at idle (no thrust) until thrust is turned on.
The trajectory evolves in the capture basin $\mathrm{Capt}_{(3.2)}(K_{\mathrm{flare}}, C_{\mathrm{touch\ down}})$ until a point A_1 where it hits the boundary of that set. The corresponding history of the flight parameters can be seen in Fig. 3.7. As can be seen from this figure, until the trajectory reaches the point A_1, the aircraft is in idle mode (zero thrust), until thrust is turned back on to avoid leaving $\mathrm{Capt}_{(3.2)}(K_{\mathrm{flare}}, C_{\mathrm{touch\ down}})$ (which would lead to an unsafe landing, i.e. the impossibility of reaching the target while staying in the constraint set). As can be seen from the other flight parameters, speed is almost constant during the landing, altitude decreases with a plateau (corresponding to a zero flight path angle). The sharpest descent happens at the end around A_1. As can be seen from Fig. 3.7 right, as soon as the trajectory hits A_1, it stays along the boundary of the capture basin, i.e. viability is ensured by applying the viability feedback (shown in Fig. 3.7).

2. **Trajectory emanating from** B.

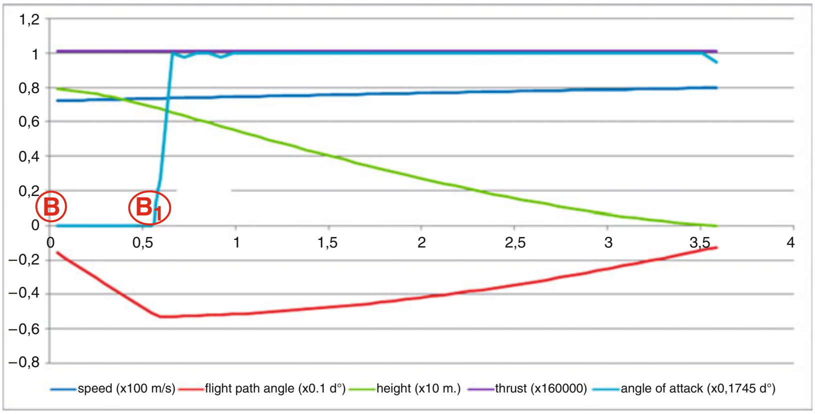


Fig. 3.9 Graph of the evolution starting from B.
Evolutions of the flight parameters and the control variables for the landing starting from state B.

We then describe the trajectory emanating from the point B, visible in Fig. 3.7 right. This corresponds to a descent at constant thrust but zero angle of attack until the angle of attack is raised (flare) to achieve a safe landing. Starting from point B, the trajectory reaches the boundary of $\mathrm{Capt}_{(3.2)}(K_{\text{flare}}, C_{\text{touch down}})$ at point B_1, where it is forced to increase the angle of attack (which will in turn increase lift), to be able to reach the target safely. As can be seen from the temporal histories of the flight parameters (see Fig. 3.9, at B_1, the angle of attack increases, leading to a reduction of the decrease of the flight path angle. This is relatively intuitive, the increased lift due to a higher angle of attack decreases the rate at which the flight path angle decreases (hence after $t = 0.5$, one can see that the flight path angle starts to increase again. The trajectory reaches the target along the boundary of the capture basin, as can be seen from the temporal histories, with a slight increase in speed, and with a monotonic decay in altitude.

These two examples illustrate the power of the viability method used, which is able to adapt the feedback (by varying the thrust and the angle of attack) based on the scenario, in order to preserve viability while descending in the constraint set.

3.4 Path Planing for a Light Autonomous Underwater Vehicle

This section presents the results of the application of the capture basin algorithm to a minimal time problem (see Sect. 4.3, p.132). We are interested in finding the fastest way for a Light Autonomous Underwater Vehicle (LAUV) to reach a target location.

3.4.1 Dynamics, Environment and Target

We treat the submarine as a three degree of freedom planar vehicle. The vehicle has a propeller for longitudinal actuation and fins for lateral and vertical actuation. However, we decouple the actuators and use the vertical actuation only to maintain a constant depth.

We characterize the state of the system with three variables: x, y for earth-fixed East and North coordinates, and ψ for earth-fixed heading. For this application it is acceptable to assume that the effect of currents can be captured by superposition; in other words, the velocity of the surrounding water can simply be added to the velocity of the vehicle due to actuation.

The LAUV is modelled by the tree-dimensional system

$$\begin{cases} (i) \quad x'(t) = u(t)\cos(\psi(t)) + v_{\mathrm{cx}}(x(t), y(t)) \\ (ii) \ \ y'(t) = u(t)\sin(\psi(t)) + v_{\mathrm{cy}}(x(t), y(t)) \\ (iii) \ \psi'(t) \in [-r_{\max}, r_{\max}] \end{cases} \tag{3.7}$$

where $v_{\mathrm{cx}}(x, y)$ and $v_{\mathrm{cy}}(x, y)$ are the components of the water velocity.

The deployment area was a 400×300 m rectangle containing the junction of the Sacramento River and the Georgianna Slough in California, USA. Under normal conditions, water flows from North to South, at speeds ranging from 0.5 to 1.5 m/s.

Bathymetric data for the region is available. The channel depth drops steeply from the bank, and is deeper than 2 m at all points away from the shore, so operations can be safely conducted as long as the LAUV does not come within 5 m of the shore.

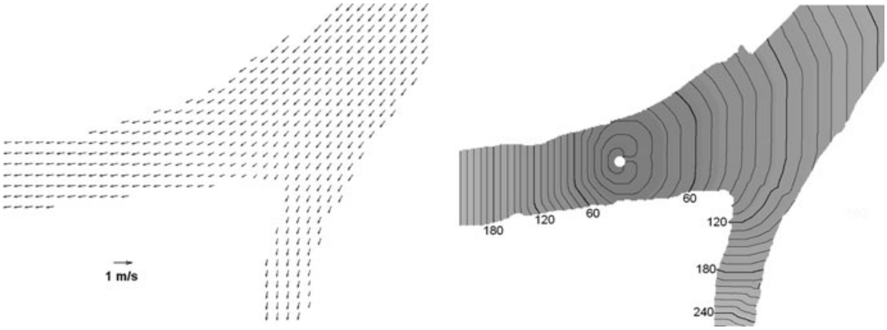


Fig. 3.10 Current Map and Isochrones of the Minimal Time Function.
The left subfigure *displays the currents in the Sacramento river and the Georgianna Slough where the LAUV was deployed for the test. The current is used in the computation of minimum time function, which itself serves to compute the optimal trajectory. Isochrones of the minimal time functions for various starting positions and initial direction East are displayed in the* right subfigure. *Numbered contours give time to target in seconds.*

Figure 3.10, p.120 shows the results of the minimal time computation using the capture basin algorithm for the LAUV model. The effect of the anisotropic current on the minimal time function is clearly visible in the figure showing the isochrones, and the presence of discontinuities in the feedback map is apparent in the eight subfigures showing the retroaction below (see Fig. 3.12, p.122).

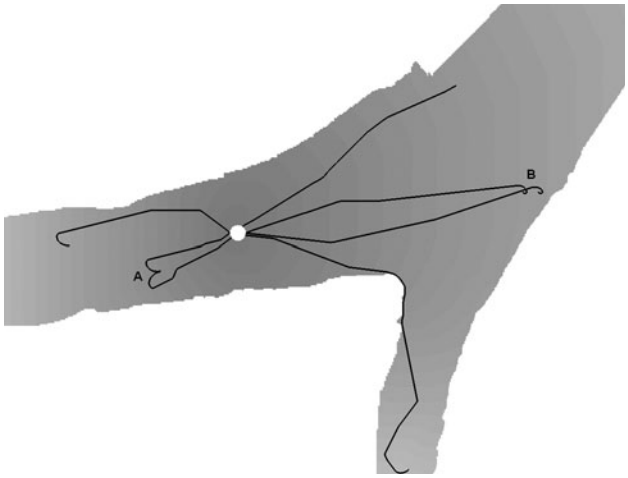


Fig. 3.11 Sample of generated trajectories.
Sample of generated trajectories. Pair "A" have the same starting position, directions differ by 15°. Pair "B" have the same starting direction, positions separated by 10 m.

In the figure above, the trajectories labelled "A" show how a change in initial direction (270° versus 255°) can result in dramatically different actions. The trajectories labelled "B" show how two starting points, separated by 10m, with the same initial direction 90°, can take very different paths to the target.

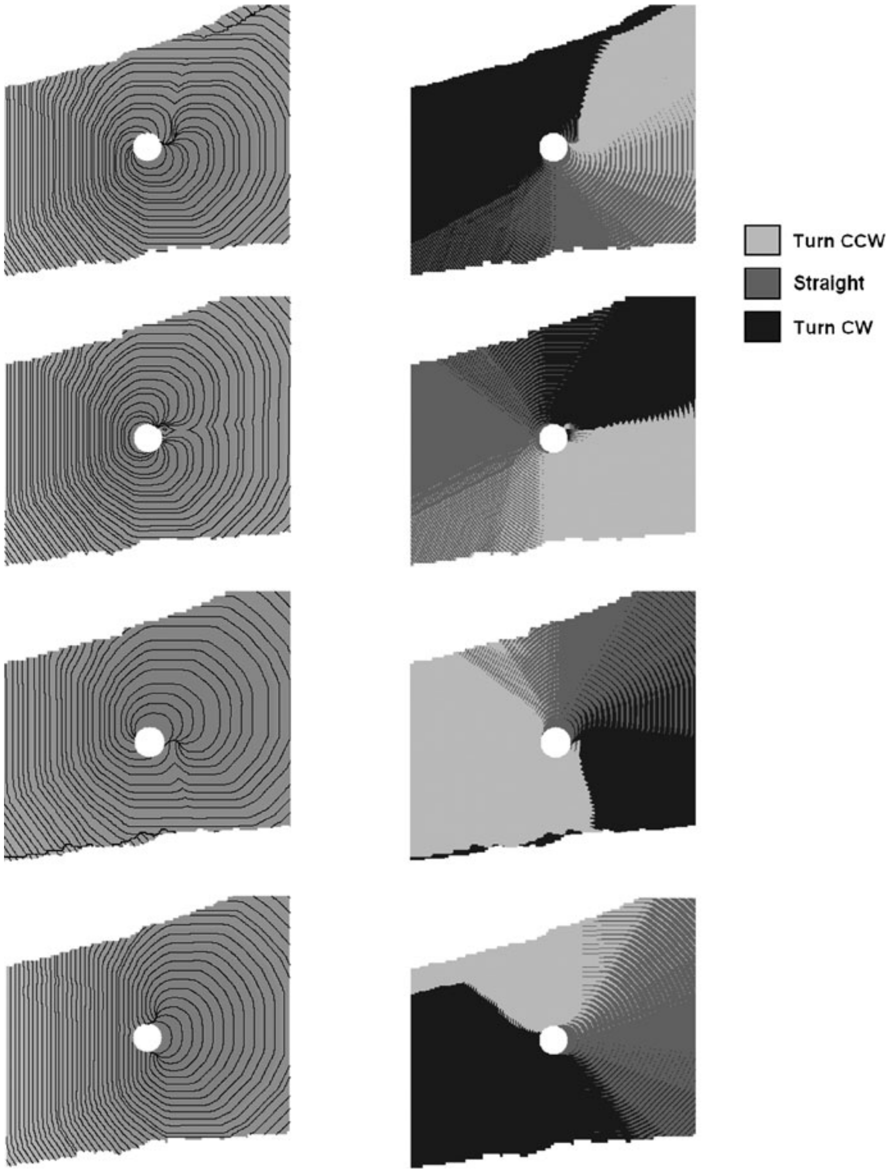


Fig. 3.12 Minimal time function (*left*) and feedback map (*right*).
*Minimal time function (*left*) and feedback map (*right*) for a 100 × 100 m region around the target, and initial direction (from* top*) North, East, South,*

West. The direction feedback map indicates directions of heading change (CW stands for clockwise *and CCW for* counterclockwise*).*

3.4.2 Experimental Results

For the experiment, we use the following values which result from our tests. The maximum speed of the submarine in still water was experimentally determined to be 2 m/s, and at that speed, the maximum turn rate was 0.5 rad/s. By fitting a hydrodynamic model to these values, the turn rate/speed relationship could be estimated for lower speeds. The value 2 m/s was judged to be too fast for the planned experiments, for safety and other logistical reasons. We used an intermediate value of $V = 1\,\mathrm{m/s}$ and $r_{\max} = 0.2\,\mathrm{rad/s}$.

Fig. 3.13 Light Autonomous Underwater Vehicle (LAUV).
The Light Autonomous Underwater Vehicle (LAUV) from Porto University used for the implementation of the algorithm. The LAUV is a small 110 × 16 cm low-cost submarine with a maximum operating depth of 50 m for oceanographic and environmental surveys designed and built at Porto University. It has one propeller and three control fins. The onboard navigation suite includes a Microstrain low-cost inertial measurement unit, a Honeywell depth sensor, a GPS unit and a transponder for acoustic positioning (GPS does not work underwater). This transponder is also used for receiving basic commands from the operator. The LAUV has a miniaturized computer system running modular controllers on a real-time Linux kernel.

The viability algorithm returns the minimum time function and the feedback map for the chosen target. Due to the way the minimum time function is built, it is efficient to calculate the feedback map at the same time. The feedback map is a function that returns the direction command necessary to stay on the optimal trajectory at each position/direction in the viable set. Ideally, the LAUV would use this feedback map directly. In our experiment, we instead use this feedback map to pre-calculate the optimal trajectory. This is done by selecting a starting point and direction, then finding the optimal trajectory from that point by using the feedback map to find successive points. In other words, there is no need to perform a dynamic programming optimization on the minimum time function; the feedback map made the trajectory calculation straightforward.

We ran several experiments with the trajectory data sets provided by the optimal control algorithm. This took place in second week of November 2007. The qualitative behavior of the LAUV did not change significantly across several experiments, and showed good agreement with predicted results.

Figure 3.14, p.123 displays the results for the experiment involving the optimal trajectory planning. In this experiment the LAUV was deployed at the location labelled *start*, where it drifted with the current while waiting for the *startmission* command.

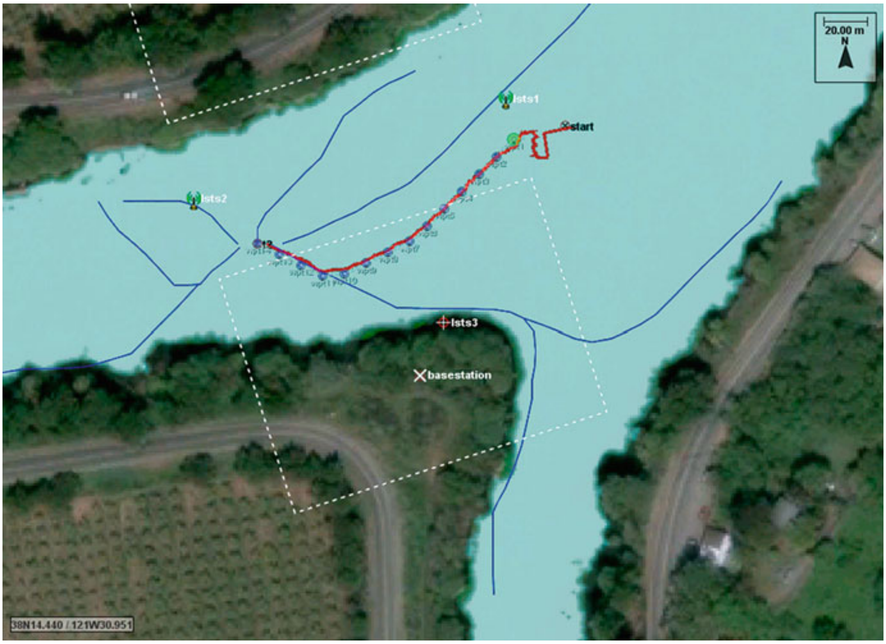


Fig. 3.14 Minimal time evolution of the submarine.
The minimal time evolutions of the submarine are computed by the capture basin algorithm viable in the Georgianna Slough. The submarine starts from the start point and rallies the target following the optimal trajectory obtained using the minimum time function.

Figure 3.15, p.124 shows the deviation between the planned and actual trajectory.

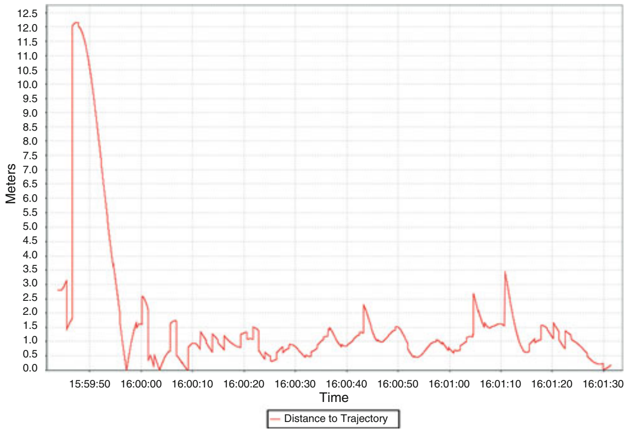


Fig. 3.15 Deviation between planned and actual trajectory.
The deviation between planned and actual trajectory is obtained by comparing the estimated position of the submarine during the operation (by acoustic methods) with the planned position obtained from the viability algorithm.