FLIGHT TRANSPORTATION LABORATORY REPORT R 89-1

TERMINAL AREA FLIGHT PATH GENERATION USING PARALLEL CONSTRAINT PROPAGATION

Michel Marc Sadoune

May 1989

Abstract

A Flight Path Generator is defined as the module of an automated Air Traffic Control system which plans aircraft trajectories in the terminal area with respect to operational constraints. The flight path plans have to be feasible and must not violate separation criteria.

The problem of terminal area trajectory planning is structured by putting the emphasis on knowledge representation and air-space organization. A well-defined and expressive semantics relying on the use of flexible patterns is designed to represent aircraft motion and flight paths. These patterns are defined so as to minimize the need for replanning and to smoothly accommodate operational deviations.

Flight paths are specified by an accumulation of constraints. A parallel, asynchronous implementation of a computational model based on the propagation of constraints provides mechanisms to efficiently build feasible flight path plans.

A methodology for a fast and robust conflict detection between flight path plans is introduced. It is based on a cascaded filtering of the stream of feasible flight paths and combines the benefits of a symbolic representation and of numerical computation with a high degree of parallelism.

The Flight Path Generator is designed with the goal of implementing a portable and evolving tool which could be inserted in controllers' routine with minimum disruption of present procedures. Eventually, a model of aircraft interaction provides a framework to rethink the notion of Separation Standards.

Contents

List of Figures

Chapter 1

Introduction

This introductory chapter first highlights the tensions which strain today's airport and air-space system and presents a set of complementary measures to increase the capacity of the system with different time perspectives of implementation. The introduction converges to an overview of the problems involved in the planning and generation of flight path plans in the terminal area as a contribution to the improvement of the present Air Traffic Control system. After a literature review, the approach to the problem of path generation of this report is described by putting the emphasis on the incremental design and development of a software tool.

1.1 Future of the Aviation System

The fast growth of civil aviation activity in the past recent years has severely stressed the existing airport and air-space systems. Congestion and significant delay are commonly experienced at many of the busiest airports throughout the United States and in the world. The tensions which appear today may be signals of a rising crisis in aviation. [15]

1.1.1 Expected Growth of Aviation Activity

Air travel is the main component of the increase of the aviation activity [1]. The growth of air travel demand is closely related to demographic trends, the state of the economy, changes in life-style. Drawing on econometric models under various scenarios compounding these parameters, demand in air travel expressed in revenue passenger miles, may be forecast to rise by at least twice and possibly six times by the middle of the next century. The increase in air traffic activity will

[1] During the last decade air travel in the United States has increased by 75 percent reaching the volume of 468 million passenger enplanments in 1987 [13].

depend on the nature of the airlines' response at accommodating this expanding market [6].

1.1.2 Concentration of Traffic

The structure of the airport network reflects a progressive concentration of traffic at a few major airport which serve as connecting points, or hubs for converging traffic routes. Airlines favor hubbing as an efficient operating pattern which allows them to increase flight frequency and load factor. They have been using it increasingly since deregulation. In the United States the ten top airports handle forty percent of the traffic [39]. Local congestions propagate in the form of delay from these key airports to the entire network. Capacity at these nodes of intense traffic has been stretched to practical limits. The ability of the airport network to accommodate expected increase in air traffic over the coming years depends on traffic handling at the main airports where traffic has been concentrated.

1.1.3 Improvements in Air Traffic Control

Measures consisting of upgrading and enhancing the present system probably involve the least cost and disruption. They can be implemented incrementally and start delivering improvement in the short-term. To provide more capacity at the existing airports one may first try to get the most out of the existing system by the means of selective constructions and technological or procedural improvements. The opportunity to expand an existing airport by adding new runways or taxiways, improving terminal facilities and ground access is site-dependent. New navigation equipments and improved Air Traffic Control facilities would increase aircraft controlability and would give to the controllers the means to organize traffic more efficiently with increased safety. For instance traffic flow can be increased by procedural and operational changes in the system. Improved traffic-handling procedures may be devised to smooth demand peaks and reduce delays instead of controversial economic measures such as differential or peak-hour pricing. Scheduling runway operations would increase the runway throughput. Indirectly, better information about the elements influencing aircraft controllability would allow reducing or adapting separation criteria. In a longer term the aircraft with 4D terminal area navigation equipments will be able to conform with a planned trajectory. The flight path plans will have to meet the runway schedule and avoid conflicts with other aircraft. The *FAA*'s airport capacity enhancement plans [1] harmonize the scheduling of the efforts for the development of Air Traffic Control technologies and procedures. The contribution of these local improvements to the global system are difficult to appreciate.

1.1.4 Enlarging the Infrastructures

In the long term air activity will undoubtly overwhelm the capabilities of the present airport system. Optimizing the use of the present infrastructures will provide only temporary relief. Large-scale measures to increase the fundamental capacity of the airport system will eventually become mandatory. One may remark that airport development which involves major capital investments on often controversial construction projects happens to be usually deferred and lags behind the growth of aviation activity need instead of anticipating it.

The airport network will have to be enlarged to satisfy the rising demand for air travel. Adding new airports in the major metropolitan areas which handle the most traffic would significantly increase the volume of traffic that can be accommodated at these areas and throughout the network. This concentration of ground facilities may actually increase air-space congestion but it would avoid the saturation of the airport system. Air traffic control would become even more challenging a task especially if air-space tends to be the factor limiting capacity. Aircraft flow control, air-space management and trajectory planning will have a crucial role to efficiently use the available air-space.

Building new airports in these areas however is limited by the lack of suitable site, the environmental impact, the adverse effects of aviation activity on surrounding communities. Conversion or joint-use of general aviation and military facilities to air carrier operations may encounter the resistance of users.

1.1.5 Complementary Approaches

Other more innovative solutions may be envisioned in a longer term perspective. Recognizing the pressure of connecting flights on major hub airports, this part of the traffic could be decentralized and spread amongst many more existing airports. Advances in vehicle technology may allow an opposite approach. Ground facilities for vertical take-off and landing aircraft ($VTOL$) could be integrated in urban areas with additional noise protections and quieter engines. $VTOL$'s would operate on the shortest flights from city to city. Larger aircraft could transport more passengers on traditionally busy routes. This would not increase the number of operations and only minimal adaptations of present airports would be necessary.

1.2 Overview of the Problem of Flight Path Generation in Terminal Area

Air traffic congestion and frequent saturation of major airports are issues which question current Air Traffic Control (ATC) system and procedures. Airports are often a bottleneck in any attempt to increase the flow of air traffic. Busy airports may reach a state of saturation at peak hours and may not be able to

absorb the arriving flow of traffic or scheduled departures. Excessive delays result in passenger discomfort, fuel waste and disruption of airlines schedules. For such economic reasons, a goal of *ATC* is to ensure an expeditious movement of aircraft. Safety remains nevertheless the primary concern and should not be compromised when trying to accommodate an increasing flow of aircraft. More efficient Air Traffic Control procedures in terminal area are a potential source of increase in system capacity. Efforts are invested to serve this purpose in the design and development of automated decision support systems for terminal area air traffic controllers.

Operation Research techniques can be successfully applied to plan air-traffic activity by optimizing the flows of traffic and the sequencing of operations between the various parts of the system. The resulting schedules constrain aircraft motion but much freedom remains available for choosing and explicitly defining the trajectories which are actually flown by the aircraft. Coordinating aircraft motion in space by a proper planning of flight paths should allow a more efficient use of the available air-space and an improvement of *ATC* procedures should be gained from it. Regulating the flow of traffic, finding a set of flight paths which are free of all violation of safety regulation, guiding the aircraft in conformance with the planned paths and maybe modifying the paths are issues involved in the process of flight path plan generation. Advances in the technologies of surveillance, navigation and communication now allow an increased aircraft controllability that should make trajectory planning feasible.

Whereas scheduling the traffic for airport and *ATC* facilities is amenable to a mathematical optimization or heuristic method, no formal model is available to organize the air-space and choose the aircraft trajectories. A main difficulty involved in the process of choosing a flight path plan rests in the essentially continuous nature of space and time that allows an infinity of possible solutions and in the variety and complexity of the constraints imposed on the trajectories.

This research investigates the problems involved in the planning and generation of flight path plans in the terminal area and describes a software tool which has been developed to aid controllers achieve this task.

1.3 Literature Review

A brief literature review is aimed at pointing out the successive approaches to the problem of flight path generation in terminal area.

Early attempts to generate flight path plans were single-aircraft oriented. They focussed on the problem of finding a path which could be flown by a given aircraft and relied on optimality conditions to master the infinity of possible paths. Horizontal geometry of the path, vertical profile and speed profile were dissociated and these three components treated separately. The vertical profile could

be optimized according to aircraft capability or fuel-consumption criteria. Time constraints could be satisfied by speed control. On the other hand choosing a horizontal path remained an unstructured problem and the main difficulty.

The first step consisted of finding a geometric path between two points in a two-dimensional space when heading constraints could be specified at the extremities. The guidance law minimized the arc length of the path subject to the constraints of a minimum turning radius of the aircraft. Intuitively the shortest path when the points are sufficiently distant is composed of a straight-line segment with minimum radius turns at the extremities. More generally, an optimal path should consist of a sequence of minimum radius turns and straight-line segments. [10]

This result was extended in [12]. An approach route was defined as a sequence of way-points through which the aircraft was to fly consecutively. The shape of the horizontal component of the trajectory was synthesised from straight-line segments and circular arcs. Two consecutive points were connected by a straight-line and heading adjustments at the extremities ensured smooth transitions. A speed profile and a vertical profile had to be superimposed on the horizontal path. Again, the selection criterion was the minimization of the total length of the path. If the time of arrival could not be achieved by speed control the path could be stretched by a lateral deformation. This means of representation of a flight path complied with the main aircraft maneuvering constraints and was attractively simple. However its expressive power was restrained by the unnecessary imposition of fixed way-points. The notion of optimality which was used to help constraining the shape of the path lead to generating a unique path independently from other traffic.

When multiple aircraft are considered, conflict avoidance requires ensuring the respect of *ATC* minimal separation criteria at all times.

The control of multiple aircraft was investigated for an aircraft carrier landing system where aircraft were all fighter-type with similar velocity characteristics. The main purpose, however, was to optimize the flight path of each aircraft and conflicts had to be handled empirically. Optimal trajectory segments for a single aircraft, such as minimal-fuel landing, were introduced as portions of the path. [31]

A flow control approach was adopted to find an optimal solution for the problem of landing a merging string of vehicles while maintaining adequate separation between aircraft. After the case of a single aircraft category and a single runway [38] [4], a mix of aircraft and multiple runways were considered [40]. The terminal area was modeled as a set of nodes: entry points, outer markers, and intermediate points such as path intersection points, holding and reporting points. A unique nominal air route structure was specified horizontally and vertically from the entry points to the outer markers. It was assumed that the detail of the trajectories along the routes between adjacent nodes could be determined independently by a method such as the one above, using straight-lines and portions of circles. Time

slots during which each aircraft could be scheduled at a node were adjusted iteratively so as to avoid conflicts. Scheduling an aircraft from an entry point to a runway meant specifying for each node in the path a conflict-free time at which the aircraft was to pass through the node. Using the nodes as check points during the progression towards a runway would have improved conformance with the path plan to meet a landing schedule. On the other hand the path might have to be stretched locally to satisfy intermediate time constraints. Conflicts were avoided by delaying aircraft in holding patterns. Avoiding conflicts between nodes involved inaccurate extrapolations from the conditions at the nodes. This approach had other more fundamental drawbacks. Building a trajectory in isolation between two nodes did not allow a global planning. The freedom in the choice of the flight paths was lost by constraining the aircraft to follow a common route indifferently from their flight capabilities.

Optimal control techniques have been used to build approach flight paths for a system of aircraft. Paths were curved to take advantage of the freedom of trajectory selection that could be exploited by the use of the Microwave Landing System (MLS). Flight paths were governed by aircraft equations of motion. In a first approach, the performance criterion to minimize was the sum of flight durations plus an integral function of aircraft accelerations in the terminal area so as to obtain smooth trajectories [30]. Separation inequality constraints were imposed between the trajectories. Additional aircraft performance capability constraints would have been necessary to ensure that the paths could actually be flown. The complexity of the optimization algorithm would have restricted its practical use to a very small number of aircraft. Another optimal control approach consisted of minimizing an integral function of the angular speed [17]. The computation time was reduced by considering the aircraft one after the other, but the iterative optimization method was sensitive to convergence issues. Four-dimensional flight paths were generated for the final precision delivery stage and could be stretched to meet a landing schedule. These paths were intended to be used as nominal paths for all aircraft. Applying optimal control methods to the generation of flight path plans had inherent limitations. The resulting paths were only defined numerically. Generating a single optimal plan did not allow any flexibility in air-space organization. The changing curvature of the trajectories required MLS coverage and an advanced degree of automation of terminal area operations to generate and follow the paths.

The Metering and Spacing program sponsored by the Federal Aviation Administration was developed in an effort to provide some automation to increase airport landing capacity. For the first time an automated flight path generator was accomplished but it lacked conflict detection and the controller was required to provide altitude separation manually to resolve conflicts. The system had to be reinitialized after each controller intervention. [37]

An original approach, which raised more interest in the field of Artificial Intelligence than for its applicability in *ATC*, was a distributed planner for air traffic control. The planner relies on mainly negative constraints to avoid conflicts and each aircraft entity is responsible for finding a path by communicating with the rest of the traffic. [75]

Presently under development by the US Marine Corps, *MATCALS* (Marine Air Traffic Control And Landing System) is intended to be a deployable system designed for existing Marine Air Traffic Control Squadrons. It should upgrade and automate the terminal area air traffic control and all-weather landing systems. The *MATCALS* terminal area *ATC* system specifies both datalinking cross pointer display information to guide the pilot on a complex path, and also an automatic mode which sends computer-generated ground command directly to the Automatic Flight Control System of modern navy tactical aircraft. Marine ground controllers will be able to draw desired two-dimensional terminal area flight paths on their display screen with a light pen and specify altitudes at way-points whereas the pilot retains control over throttles and airspeed or may disengage his aircraft from the automatic control. [43] [14] These attractive tools should allow controllers to specify and visualize flight paths in advance but no automated logics is provided to generate flight path plans for terminal area from a strategic point of view. The communication and ergonomic facilities to be developed should contribute to the future automation of terminal area *ATC* system.

A prototype of an Expert System was previously developed at the MIT Flight Transportation Laboratory to generate terminal area flight path plans for arriving aircraft and could be easily extended to deal with any kind of traffic. Several Artificial Intelligence techniques were combined to design a planner based on partially predefined sequences of actions involving mathematical descriptions such as movement in space with a time requirement. Emphasis was put on the expressivity of the navigability constraints imposed on the flight paths that fit in a global organization of the terminal area. A methodology to deal with potential air-space violations was suggested that would take advantage of a symbolic representation of flight paths by combining analytic resolution and empirical rules, but no conflict detection function was implemented at that time. [25] Suggestions for the integration of this approach in the terminal area *ATC* operations and for conflict detection and resolution appeared in [33].

An advisory system is currently developed at NASA Ames Research Center using an Expert System approach to predict and resolve air-space violations. The purpose is essentially to be a decision-aid tool to advise controllers in vectoring aircraft when a conflict has been detected. The trajectories are extrapolated up to a look-ahead time by the use of four-dimensional guidance algorithms. Potential conflicts are detected and resolved locally and sequentially. Three rules and some simple geometric considerations help decrease the computation required for

a point-to-point separation testing. Only a tactical view of the terminal area activity is considered and conflicts are resolved "step by step". The reasoning involved is highly dependent on the traditional but restrictive use of standard paths along which aircraft are in trail. [21]

1.4 Approach to the Problem

A functional description of the future *ATC* operations in the terminal area appears in chapter 2. A Flight Path generator is described as a module of an automated *ATC* system which coordinates the motion of aircraft in the terminal area with strategic purposes by generating conflict-free feasible flight path plans. This thesis proposes to design and implement a prototype of such a Flight Path Generator. Emphasis is put on the evolution from tactical and empirical procedures towards a strategic organization of automated operations without disruption of current practice.

A first prototype of a Flight Path Generator had been designed in the form of an Expert System [25]. It made use of, and adapted, several Artificial Intelligence techniques that were coordinated by a knowledge-based system. A difficulty was the lack of formalized body of knowledge dealing with spatio-temporal reasoning. A formalization of aircraft motion in space and a representational framework of flight path plans was introduced to address the controllers' needs and to allow a symbolic manipulation of trajectories. A high-level description allowed reasoning in terms of aircraft maneuvers and controller vector commands, rather than in terms of mathematical relations.

These means of representation have been extended and refined to provide more expressive power and facilitate the construction of more flexible flight path plans. The plans can be described and manipulated with different degrees of abstraction which are organized in a hierarchy. At the highest level of abstraction, the use of patterns is a key-stone of the representational model of flight path plans. It allows one to describe, efficiently build and modify flight path plans by combining the constraints of various nature which are imposed on the path. Basically, a pattern is a predefined type of flight path with degrees of freedom which can be adjusted to form various paths with similar properties.

The knowledge-based system was implemented as a rule-based system using the logic programming language Prolog. The embedded knowledge captured the relevant aspects of kinematics, geometry and mechanics within the limit of the representational framework. The knowledge base anticipated situations which might arise by using a set of flexible patterns. The system relied on some compiled knowledge for its reasoning power. Empirical rules had been given the priority over a model-based reasoning. A detailed model of aircraft dynamics and of *ATC* procedures was part of the terminal area simulation to which the Flight Path Generator was to be coupled. This model defines what will be described here as an

operationally "feasible" flight path plan. Equation solving methods, a propagation mechanism and a numerical pattern matcher were used to incrementally build plans from flight path patterns.

This implementation had the advantage of being extremely flexible. The characteristics of the generated flight paths could be easily modified. Patterns could be designed and tested at low cost with minimal changes in the code. The Expert System was used as a development tool to improve the description of the flight paths, to design new patterns and to try alternative construction strategies. It allowed an increase in the expressive power of the representational framework and the adaptation of the construction mechanisms.

In spite of these attractive properties the first implementation suffered from an important drawback. To be of help to the controller, a multiplicity of flight paths should be generated by the Expert System during his decision time. This real-time requirement was not satisfied. Optimizing the code, moving to a faster implementation of Prolog, to another programming language or to faster hardware would have made this implementation viable if one had not expected to be confronted with the same problem again after some necessary extensions to incorporate a conflict identification and resolution function.

In a multi-aircraft environment the generated flight path plans should be not only feasible but also conflict-free with respect to current *ATC* Separation Standards. Given a distribution of aircraft in space with initial positions and kinematic conditions, the problem of flight path plan generation consists of finding a set of conflict-free paths through the terminal area which accommodates each of the aircraft. Arriving aircraft have to be lead from terminal area entry points to a runway and departing aircraft from take-off to en-route corridors. Though the number of aircraft which may be simultaneously present or expected in a short-term in a terminal area is relatively small, handling them efficiently requires a well-defined methodology to deal with the complexity of planning in a multi-agent domain.

Assume the problem of finding a set of conflict-free paths has been previously solved for N aircraft. When a new aircraft is expected to enter the terminal area air-space, the problem has to be solved with the additional new aircraft. Since the problem is similar to the previous one it could be solved again independently. However in addition to the computational expense involved in generating a completely new set of paths, a constantly changing image of the activity is unacceptable for a controller.

A sequential approach takes advantage of the already existing plans. Paths for the new aircraft are generated in their order of arrival, while the prior set of conflict-free path plans remains fixed. The previously generated paths and the volume surrounding them to ensure separation criteria are then considered as moving obstacles which must be avoided by any new trajectory. This conservative approach has the advantage to cut the computational expense by avoiding unnecessary replanning of prior paths. However if it happens that no new path can be

found, in the sense that no satisfactory solution can be expressed or generated, a deadlock may be avoided by relaxing the commitment to prior planning. Paths may be undone for a limited number of conflicting aircraft and new ones generated. By relying on the capability of the Flight Path Generator at providing a multiplicity of alternative paths with various characteristics, finding conflict-free paths should be possible in most realistic situations since the critical air-space restrictions apply only to landing aircraft at final approach.

The generation of feasible conflict-free flight path plans for a given aircraft while the rest of the traffic is abstracted as a set of obstacles, is modeled as a generate-and-test heuristic. The basic principle is that a first module generates candidate paths which are tested by a second module so as to keep only solutions to the problem.

The generator delineates the set of all plausible paths that will be allowed as solution candidates. Because of the infinite number of envisionable trajectories, one may fear that a generator which is too prolific and tends towards producing a plethora of irrelevant paths would make testing untractable. A solution to this design problem is to strengthen the heuristic control over the generation of candidate paths by constraining the form of the trajectories which can be generated. At the other extreme a very "smart" generator would alleviate the need for testing if all generated paths were perfectly feasible and conflict-free. The structure of such a generator would intermingle the notions of feasibility of a flight path and of conflict avoidance. It may not be portable to adapt to the specificity of each terminal area and may be difficult to evolve with *ATC* procedures and aircraft technology.

There exists a continuum between these two extremes and the design choice should strike a reasonable balance for an efficient heuristic. A meta-level for preplanning or some feedback from the test module may help direct the generation process. The design and implementation of the Flight Path Generator are based on the following choices. An expressive and easily adaptable representational framework constrains the definition of feasible flight path plans. Such paths are efficiently generated by a parallel construction mechanism. A combination of conflict detection algorithms are used for fast testing.

The construction method of feasible flight path plans has been derived from the previous implementation. It was noticed that the time requirement of the trajectory generation process depended mainly on the efficiency of the propagation mechanism. This underlying mechanism actually constructed flight path plans in the following way: whereas constraints were applied on the initial pattern, their effects were incrementally propagated until an entirely specified flight path plan was defined as the result of this accumulation of constraints. The choice of a propagation mechanism determined the order in which the different parts of the path were built and consequently influenced the speed of construction.

A computational model based on the notion of constraint highlights the par-

allelism and other fundamental properties which are inherent to the definition of simultaneous constraints and the propagation of their effects. The dataflow model of computation shares some of these properties. An implementation of this model with dataflow architecture computers provides an efficient tool for building flight path plans. A wide variety of feasible paths can then be generated in parallel in a minimal time, to be subsequently tested for conflict and displayed to the controller.

The test module ensures that the generated paths are conflict-free. Multiple conflict detection algorithms are organized in layers which reflect the hierarchical description of the flight path plans. They combine empirical rules based on properties of the patterns, symbolic properties of geometry and kinematics, mathematical algorithms in an analytic form and numerical simulations. The development of rules and algorithms to detect conflict showed that an important amount of parallelism is inherent to the problem of conflict avoidance. Parallelism appears not only when candidate paths are tested against a set of prior paths but also at the level of a given pair. Conflict detection also benefits in time-efficiency from the fined-grained parallelism of dataflow programs.

The Flight Path Generator is then able to provide the controller with a choice of conflict-free feasible flight path plans.

The possible exploitation of parallelism to build flight path plans and to deal with potential conflicts suggest a drastic change in the approach to the problem of flight path generation: the use of parallel computers and the design of parallel algorithms, which should at the same time eliminate the concerns about speed limitations.

A direct jump to a parallel implementation however would hardly have been achievable. A first prototyping in the flexible form of an Expert System using a logic programming language allows one to define the structure of the problem from a computational point of view and to test the tractability of the representational model.

The computing facilities which are available for this project include an integrated programming environment called *Id World*, which runs on workstations such as Lisp machines, and provides tools to develop parallel programs in the *Id* language and to run them by emulating a dataflow architecture computer. Statistical data about a program execution, such as instruction counts and parallelism profiles, can be automatically gathered and plotted. The same programs can be run without recompilation on the *Multiprocessor Emulation Facility* which is a collection of thirty two Lisp machines interconnected by a network so that they can cooperate in a distributed manner. Also, the prototype of a dataflow architecture computer is expected to be available at the *MIT Laboratory for Computer Science* by the time of completion of the project presented in this report.

Chapter 2

Functional Description of *ATC* Operations in the Terminal Area

This chapter introduces the issue of flight path generation in the context of Air Traffic Control (*ATC*). After defining the operational variables involved in the control of air traffic, a functional description of *ATC* operations in the terminal area exhibits the role of a Flight Path Generator. Emphasis is put on the evolution from tactical and empirical procedures towards a strategic organization of automated operations without disruption of current practice.

2.1 Definition of the Operational Variables

The system composed of a ground controller supervising multiple aircraft involves constant interactions between its elements. Measurements characterizing these interactions come from various sources and are exploited in different locations. The variables involved in the description of the state of an aircraft and in the identification of air-space violations, as they are perceived by different parts of the system, are defined below. [32]

2.1.1 States

The state of an aircraft characterizes the aircraft at a given time. It includes its position and relevant kinematic flight characteristics.

- S_i : actual state of aircraft i.
- $\hat{S}_i^{ground}$: estimate of S_i seen by the ground.
- $\hat{S}_i^{on-board}$: estimate of S_i available to the pilot and airborne equipments.

- P_i : planned state of aircraft i. It is a projection of the state of the aircraft at a future time planned by a centralized ground system.

The estimates are based on a combination of ground and on-board measurements. Information comes from radar data which are refreshed with a sampling rate of a few seconds, from tracking data which are the result of a smoothing process of sampled data, from measurements performed by airborne navigation instruments usually instantaneously and with a better accuracy than measurements from the ground. These data may be up-linked or down-linked. Nevertheless $\hat{S}_i^{ground}$ and $\hat{S}_i^{on-board}$ may differ because of a lack of communication.

A flight path plan is a set of planned states for a time interval I in the future or between two events:

$$F_{iI} = \{ P_i(t),\ t \in I \}$$

The plan may be defined only over a given time horizon; it may be partially specified in its components or not be entirely communicated to the pilot who is free to choose the values of the unspecified components.

2.1.2 Separations

The essence of ATC being to ensure the safe separation of aircraft, special attention must be given to the definition of the quantities involved in the application of Separations Standards.

- $\Delta S_{ij} = S_j - S_i$: actual separation between aircraft i and j.
- $\Delta\hat{S}_{ij}^{ground} = \hat{S}_j^{ground} - \hat{S}_i^{ground}$: estimated separation between aircraft i and j as it is seen by the ground. It is inferred by difference from the ground estimates of the corresponding states.
- $\Delta\hat{S}_{ij}^{on-board\ i\ (or\ j)}$: estimated separation between aircraft i and j as it is seen by the aircraft involved themselves. This estimate is obtained through direct measurement performed by airborne collision avoidance systems.

  Note that whereas the definitions of $\Delta\hat{S}_{ij}^{ground}$ and $\Delta\hat{S}_{ji}^{ground}$ differ only by a sign, $\Delta\hat{S}_{ij}^{on-board\ i}$ and $\Delta\hat{S}_{ij}^{on\ board\ j}$ are measured by different systems.
- $\Delta P_{ij} = P_j - P_i$: planned separation defined as the separation between the future states of the aircraft. Note that a planned separation is defined mathematically without the ambiguities of measurement.
- $\Delta PS_i = S_i - P_i$: conformance error. It is the deviation between the assigned flight path plan and the actual state of the aircraft.

- $\Delta P\hat{S}_i^{ground} = \hat{S}_i^{ground} - P_i$: the estimate of the conformance error as it is perceived by the ground consists of the deviation between the assigned flight path plan and the surveillance estimated state of the aircraft.
- $\Delta P\hat{S}_i^{on-board} = \hat{S}_i^{on-board} - P_i$: estimate of the conformance error as it is seen by the aircraft itself. It may be used by airborne navigation equipments to guide the aircraft in conformance with its assigned flight path plan.

Note that $\Delta P\hat{S}_i^{on-board}$ is defined as a difference between data defined on board the aircraft and data defined on the ground, considering that flight path plans are generated by the ground system according to ground data.

$$\Delta P\hat{S}_i^{on-board} = \Delta P\hat{S}_i^{ground} + (\hat{S}_i^{on-board} - \hat{S}_i^{ground})$$

Even in the case of ideal navigation capabilities that would minimize $\Delta P\hat{S}_i^{on-board}$, the ground may perceive a residual conformance error. The discrepancy between these two perceptions of the world should disappear if complete information is efficiently communicated between controllers and pilots, or in an advanced automation environment between ground-based and airborne computer systems.

2.1.3 Air-Space Violations

Effective or only forecast, air-space violations are called respectively hazard or conflict.

A hazard is a violation of separation criteria involving the current states of the aircraft:

$$|\Delta S_{ij}| < \Delta S_{min}$$

where the inequality notation, just as the norm or the difference notations have an intuitive meaning. ΔS_{ij} is often expressed in term of time to collision. A hazard occurs if the current separation between some aircraft drops below the threshold fixed by the Separation Standards, in some dimension which may be an horizontal distance or a vertical separation.

A conflict is an hypothetical violation of separation criteria at some future time between planned states:

$$|\Delta P_{ij}| < \Delta P_{min}$$

The execution of a conflicting flight path plan should involve a hazard if no preventive measure is taken in time. The main role of a flight path planning process is to eliminate such conflicts and the consequent occurrence of hazards.

The part of *ATC* system which is responsible for the control and management of the traffic in the terminal area may be decomposed into the following functions:

- Flight Management
- Runway Scheduling
- Flow Control
- Flight Path Generation
- Path Conformance Monitoring
- Hazard Monitoring

Each function is now described with a degree of detail which allows one to define the task of each individual module, highlights their interactions in the whole system and gives guidelines for their design.

2.2 Flight Management

The Flight Management process ensures an efficient transition between en-route corridors and terminal area. It is entirely distributed among the aircraft and relies on avionics systems with which many modern transport aircraft are equipped. These Flight Management Systems help minimize the total cost of an operation between two airports. [35]

After the horizontal component of a flight path has been specified significant savings can be obtained from the choice of an efficient vertical profile. Flight Management Systems generate a reference profile for the climb, cruise and descent phase of a flight so as to minimize fuel consumption, flight time or the direct operating cost which is a combination of the two [34]. Flight Management Systems allow the pilot to plan climb and descent in a fuel-conservative manner accounting for performance characteristics of his particular aircraft and the influences of the parameters, gross weight, wind, non-standard pressure, temperature. [18] [2] [3]

Optimum profiles can be generated by the energy state method which is based on aircraft equations of motions and uses the minimum principle to optimize integral expressions [11] [20]. Slightly suboptimum but more easily applicable because they do not involve continuous changes, $Mach/CAS$ schedules (CAS stands for Calibrated Air-Speed) are speed-altitude profiles actually used by transport pilots. When descending, pilots are required to follow in a first phase a constant Mach number and in a second phase a constant indicated air-speed. The order is reversed for climbing.

$Mach/CAS$ schedules are the result of a two-parameter optimization obtained by table look-up or computed by a airborne calculator which alleviates pilot workload. Typical values appear in manufacturers' handbooks as aircraft characteristics.

In the present *ATC* system, enroute controllers who are responsible for descending arrivals for landing may receive metering advisories from a centralized automated traffic process which exists at a few airports but they are usually simply told by terminal area controllers either that holding will be necessary or that a minimum in-trail spacing should be applied to arrivals at the terminal area. Constraints imposed by the Metering Process described below can be accurately satisfied by the aircraft equipped with a Flight Management System. Descending from the Top Of Descent (*TOD*) at cruise altitude with idle-thrust in a clean configuration, landing gear up, flaps zero, speed brakes retracted, pilots are enabled to reach a metering fix at a predetermined time, altitude and speed in accordance with air traffic control requirements.

Predetermining a four-dimensional flight path plan from the *TOD* to the runway is not possible because the schedule of runway operations is not available at that time. Takeoff aircraft have not yet entered the schedule for the runway system. Besides concerns by pilots and controllers change as aircraft proceed in the descent. In the early phase of the descent as the aircraft is still in an enroute corridor and a significant altitude difference is involved, emphasis is put on the cost minimization of the flight. In the terminal area, as the aircraft gets closer to the runway in a high traffic density area, attention focuses on separations and airspace organization, accuracy of the maneuvers to meet the assigned schedule, and prelanding checks. Though a certain continuity is necessary for planning purposes the descent may be decomposed into two phases outside and inside the terminal area which are assigned to different processes: an airborne Flight Management System and a ground-based Flight Path Generator. A Metering Process allows a smooth transition by introducing time buffers to control the arrival flow of traffic.

2.3 Runway Scheduling

Considering that runway capacity is usually a limiting factor of the flow of traffic at the airports, runway scheduling plays an important role in the organization of a planned terminal area *ATC* system.

Inefficiency results from the tactical approach of today. The terminal area arrival and final spacing controllers accept the traffic flows from the terminal area entry points and execute the merge and final spacing processes with little or no recognition of current takeoff operations on the runway system. Runway controllers may ask them for extended spacing for certain peak departure periods when it is desired to insert more takeoffs in the landing traffic flow. Otherwise the landing traffic flow is established independently by the final spacing controllers. In turn runway controllers insert takeoffs into the landing flows in a tactical ad hoc fashion and are often not sensitive to any requirements from departure traffic. Departure controllers usually accept the departures as arranged by the runway controllers.

The Runway Scheduler is a strategic process which plans landing and takeoff operations on a multiple runway system. It dispatches each arriving aircraft to an operating runway and assigns a landing time to it. It also inserts take-off in the sequence of landing aircraft.

Runway scheduling can be modeled as a queuing process in which the service time depends on the sequence of the users. Indeed ATC procedures require a minimum time separation between successive landing aircraft at a runway which depends on the types of aircraft. Separation is defined according to aircraft categories to take into account runway operation time, disturbing wake-vortex effects or the possible negative influence of the presence of other aircraft on the landing system. As a consequence of the variability of the interlanding time, an optimization of the schedule by a proper reordering of the sequence of users is a combinatorial problem [9]. The objective consists either from the system point of view of maximizing the operational capacity of the runway in terms of total throughput and long term service of the runway, or from the users' point of view of minimizing the total aircraft delay while taking into account individual criteria such as the number of passengers, the fuel consumption, or the past duration of the flight.

For an aircraft to be able to meet the schedule, its assigned landing time must belong to a time interval between the earliest and the latest possible time at which the aircraft can actually reach the runway. A description of these notions and a method to get the bounds of this time interval are discussed in appendix B. Take-off should be smoothly inserted in the sequence of landings by taking advantage of gaps in the landing flow whenever possible. The Runway Scheduler produces a dynamically changing schedule as new entrants for landing and takeoff arrive and as operational deviations occur. Late modifications are not advisable because of probable irrevocable commitments of the pilot to meet the previously assigned schedule.

The commonly used First Come First Served policy is easily applicable and reflects an indisputable fairness but is not optimal. Fairness may be ensured by preventing any aircraft from being shifted by more than a specified number of positions away from its reference position in the First Come First Served sequence. Parallel algorithms deal with the runway scheduling problem as an application of the Traveling Salesman Problem involving combinatorial explosion and dynamic programming [42]. A significant increase of runway capacity and a reduction of delays can be obtained by such a mathematical optimization of aircraft ordering in the landing and take-off flows. The implementation of such techniques would push away the limits on the flow of air traffic caused by present operational runway capacity.

2.4 Flow Control

Traffic redistribution is an application of the general idea that a given volume of traffic can be accommodated with less delay for each aircraft if the fluctuations in demand have been smoothed. The aim of traffic redistribution is to avoid peaking of traffic demand at certain times or places by spreading the flow of traffic throughout the day or by diverting it to other airports serving the same area.

Flow Control is a method to redistribute traffic by achieving a more orderly and uniform distribution of aircraft movement. The Flow Control function has a strategic role which consists of managing the traffic to adapt the flows of arrivals and departures to the system capacity so as to avoid clogging any part or facility of the terminal area.

2.4.1 Metering

When metering functions are in effect, delays due to runway capacity or congestion of the final spacing air-space are experienced up-stream away from the runway.

To prevent congestion in the vicinity of the airport, the landing flow may be metered to control the number of aircraft which are simultaneously present in any sector of the terminal area air-space. Metering is a smoothing process which tries to eliminate short-term peaking in the arrival rate by causing early delays to certain aircraft in the landing stream.

Flow Control of the arriving aircraft in the terminal area occurs progressively at successive stages of the approach to the airport. The Metering Process first assigns a time to each arriving aircraft at which it is expected to reach a metering fix. These constraints can be accurately satisfied by the pilots using airborne Flight Management Systems. Holding introduces an additional time buffer to balance the flow of arrival and runway capacity. Aircraft are then metered from a gate of the terminal area or an holding point along flight paths that lead them to the runway in order to meet the landing schedule.

As far as departing aircraft are concerned, arrival at the runway for takeoff are metered by gate-hold processes. Lengthy delays can be easily accommodated in the queue at the takeoff runway unless the number of awaiting aircraft necessitates holding some of them on the ramp. The fuel cost associated with delay can be reduced by taking delays at the ramp with engines off.

2.4.2 Holding

To accommodate arrival delays, flight paths may have to be extended in time. Since there is a limit on the amount of delay which can be accommodated on airborne paths, it becomes necessary to initiate path modifications earlier or to use

holding patterns when delays get longer. Holding aircraft are stacked at holding points and isolated from other traffic. The possibility of unexpected cessation of runway operations due to weather or operational incidents will always require that holding patterns are available in all planned terminal area configuration. Reduction in ceiling and visibility or change in wind direction may involve an interruption of runway activity. As weather drops through current categories I, II, III, some aircraft can continue to land while others must be held awaiting diversion or improvement in the weather.

Metering, holding and choosing a flight path have traditionally been dissociated and tackled independently. In fact metering an aircraft from an holding point requires a knowledge of the possible paths through the terminal area to send the aircraft at the correct time for landing. For these reasons a possible holding pattern is incorporated into the arrival paths created by the Flight Path Generator described below. Once an aircraft has reached the terminal area or passed a metering fix, the amount of holding time for a given schedule depends on the Flight Path Generator.

A new Holding Stack Management function would improve the dispatching of the aircraft to the different holding points (not necessarily the nearest to the route from which they are arriving in case of congestion), and the transitions between the stack levels.

2.4.3 Risk of Runway Starvation

Metering regulates the flow of traffic in the terminal area and allows reductions of operating cost. However there is a danger of these metering processes causing unnecessary additional delay when taking existing delays further away from the runway.

Starvation would be the consequence of operational deviations which can not be compensated. If a runway goes idle due to a lack of traffic reaching it during what should be a busy period, considerable extra time and fuel costs are incurred by all subsequent aircraft of this busy period.

To take advantage of the full capacity of the runways and avoid starvation gaps in the landing flow of traffic, metering processes have the strategic responsibility of controlling the size of airborne and ground queues near the runways. To allow time adjustments not all the estimated delay should be removed early in the traffic flow. The queues should be kept small but strictly positive whenever the flow is sufficient.

2.5 Flight Path Generation

The role of the Flight Path Generation function consists of providing feasible and conflict-free flight paths through the terminal area to meet the assigned runway

schedule.

2.5.1 Standard Paths

The choice of a flight path in the terminal area traditionally relies on the use of Standard Terminal Approach Routes (*STAR*'s) and Standard Instrumented Departures (*SID*'s) for arriving and departing aircraft respectively [36]. Arriving aircraft converge from a terminal area gate or holding point down to a runway for landing. Departing aircraft after takeoff diverge up to an enroute corridor. These standard paths are published for all important airports in the world and known by pilots and controllers. They have to be strictly followed in the event of loss of communication between pilots and the ground.

STAR's and *SID*'s are three-dimensional paths defined with respect to the spatial organization of the terminal area. For each runway a set of flight paths is defined, that lead arriving aircraft from terminal area entry points to the outer marker. As they enter the terminal area aircraft are metered along these paths. Those following the same *STAR* may have to fly at similar speeds to maintain in-trail separation. In consequence of the disparity in performance of modern aircraft, different *STAR*'s may have to be reserved to accommodate slow and fast aircraft approaching the runway simultaneously. [16]

The advantages of *STAR*'s are that they are precisely predefined and indexed, they make use of navigational aids, they can be flown by any aircraft and can even be flown automatically by aircraft equipped with advanced Automatic Flight Control Systems. In practice *STAR*'s are only guidelines that let the controllers some freedom of improvisation to empirically adjust the path to each situation.

However *STAR*'s are too strict a framework to allow an efficient optimization of the air-space management. Aligning aircraft along standard paths does not take advantage of the three dimensions of the air-space to satisfy Separation Standards. Consequently using *STAR*'s may arbitrarily limit the flow of aircraft that are allowed to approach the airport, though the runway capacity may not be reached. Besides, *STAR*'s do not allow the controller to meet the requirements of the pilots, in regard to the optimum flight conditions for a particular type of aircraft.

2.5.2 Automated Flight Path Generation

The flight path generation process coordinates the motion of the aircraft in the terminal area. That requires efficiently managing the air-space with respect to *ATC* regulations and in particular Separation Standards to avoid conflicts in a multi aircraft environment.

Though flight path generation also requires some knowledge about each particular aircraft to create feasible trajectories with a concern of operating cost optimization and pilots' maneuver requirement, it is essentially a centralized ground-

based process because of traffic issues.

A Flight Path Generator would improve flow control procedures at busy airports. Unlike the current use of *STAR*'s, a Flight Path Generator should take advantage of the available air-space and of the various navigation capabilities of the arriving aircraft to provide controllers with a choice of flexible candidate paths to guide the aircraft to the runway and to deliver them at the scheduled landing time.

Given as input:

- a schedule of landing and takeoff operations for a multiple runway system
- the terminal area spatial organization and procedures
- the pertinent *ATC* separation criteria stated in terms of time, distance and altitude

and for each aircraft

- a unique identifier, such as the flight number or a call-name
- the position and the values of the kinematic parameters of the arriving aircraft
- the aircraft type and performance limitations in terms of maximum airspeed, stall speed, climb and descent rates, rate of turn, etc
- the specific maneuver characteristics expected by the pilot

the Flight Path Generator delivers as output a set of conflict-free feasible flight path plans that satisfy the following operational requirements and procedural constraints:

- plans meet the assigned schedule
- a flight path can be easily flown by the aircraft for which it has been created
- the set of flight path plans does not involve any conflict or any violation of *ATC* regulations
- the description of the flight paths must be easily understandable by controllers and pilots
- plans should allow tactical corrections indispensable in a changing world
- plans should facilitate rescheduling and replanning in the case of operational deviations

Depending on the amount of time allocated to reach the runway, the Flight Path Generator may either create a direct path from a terminal entry gate to the runway or make use of holding to absorb some delay. In the latter case it includes in the flight path a preplanned number of racetrack loops in the holding pattern at a given altitude before continuing into the terminal area air-space.

The Flight Path Generator must deal with missed approaches where an aircraft fails to land. After a minimum time desired to allow the pilot to recover safely and execute the procedure of a missed approach, a low altitude flight path is provided (which may include holding) to smoothly reinsert the aircraft into the arrival flow of traffic. During a busy period when there is a continuous stream of planned landings a missed approach introduces a gap in the arrival flow. As the Runway Scheduler adapts the sequence of runway operations to minimize a loss of capacity, new flight paths must be provided to meet the new schedule.

2.6 Path Conformance Monitoring

The Path Conformance Monitoring function supervises the execution of the flight path plans by the aircraft. It cannot be dissociated from the issues of command transmission and guidance.

2.6.1 Radar Vectoring

The common mode of exercising control over arrival and departure paths is called radar vectoring. While there is a general pattern to the paths being used there is no precisely predefined trajectory (four-dimensional path) for each aircraft. At appropriate intervals the *ATC* controller specifies a change in heading, speed or altitude by issuing a vector command by voice radio communication. Path errors arise from pilot response to these commands, from pilot conformance to the commanded values, and from wind speed and direction variations in location, altitude and time. The trajectory followed by the aircraft may not conform to the controller's vague intentions but an adjustment can always be made in issuing the next vector.

This non-precision control is essentially tactical and empirical. It is based upon a two dimensional pictorial display driven by radar and beacon surveillance data which provide position and altitude information with a few second refreshment rate. Current trackers are straight-line steady state systems incapable of providing accurate and reliable ground-speed, direction, or altitude information for roughly one minute after any commanded change in these quantities. *ATC* controllers monitor successive intervals to guess at the current direction and ground-speed of aircraft. Thus the future trajectories of aircraft are only known in a vague, general manner and actual flight paths are determined tactically when unplanned real-time situations occur.

2.6.2 Command Transmission

Flight path plan data have to be transmitted to the aircraft and translated into a set of commands which, if followed, should guide the aircraft along its assigned flight path.

Since the schedule and the associated set of conflict-free paths are dynamically changing one should not commit too early in the descent to plans which are subject to later modifications. For this reason complete plans should not be entirely up-linked as soon as they have been generated by the ground-based system but only partially for a given duration in the future which does not involve unnecessary commitment but gives sufficient advance notice to the pilot. On the other hand pilots may sometimes prefer to be informed of the current intentions of the ground even if the paths happen to be altered later.

Rather than transmitting a sequence of commands which would require some interpolation to rebuild the assigned flight path, transmitting the plan itself would have the advantage of letting the aircraft, in direct contact with the reality of the situation, tune the commands and adjust its necessary reaction to conform with the plan. Flight path plans are defined in ground coordinates and the aircraft should compensate for the effect of the noise component of the wind which can not be known at the time the plan is created. The flight path which has been assigned to an aircraft can be displayed to the pilot on a video screen with indications of altitude and speed. The expected characteristics of the maneuvers should also be explicit.

2.6.3 Conformance Monitoring

In the case of a detailed kinematic specification of the assigned flight path plans created by the Flight Path Generator, some rigor is expected in the execution of the plans. Path Conformance Monitoring is a new automation function to be added to the terminal area air traffic control processes when the paths are explicitly defined, in order to monitor aircraft adherence to their assigned flight path plan. It can work in 2D, 3D, or 4D.

The flight guidance and control system steers the aircraft such as to conform with the plan. Short-term corrections are constantly performed so as to stick to the plan by minimizing the error $\Delta P\hat{S}_i^{on-board}$. As navigation and guidance systems become more and more accurate and reliable, some modern aircraft would be able to automatically follow an up-linked flight path plan. Despite the necessary robustness of the automation, unexpected atmospheric disturbances, errors in the estimate of the wind, misunderstanding between controllers and pilots or misuse of the equipments are the main causes of conformance error.

The ground supervises the execution of the plans which have been assigned to the aircraft. When the conformance error $\Delta P\hat{S}_i^{ground}$ exceeds a given threshold a

conformance alarm is issued. The rate of conformance alarm influences controllers' workload.

Conformance resolution involves tactical measures to reduce the discrepancy between the plan and the result of its execution. Depending on the nature and the amplitude of the conformance error between the planned and the actual state of the aircraft, in horizontal position, altitude, speed or heading, there is a choice in the resolution to adopt which regains conformance with the plan. Tactical corrections which may involve additional maneuvers may be required to regain conformance with the original plan. An alternative solution, which may prove to be the only possible one in the case of a large deviation, consists of modifying the plan or even creating a new one to bring the plan into conformance with the current state of the aircraft.

2.6.4 Mixed Conformance Capabilities

The traffic flow is generally composed of a diversity of types of aircraft such as jets, turbo-props, pistons, helicopters, with varying flight guidance capabilities. The performances of the airborne equipment for navigation and guidance, the level of automation of on-board flight control systems or even pilot training are also very dissimilar. As a consequence of the disparity in aircraft controllability traffic will always involve a mix of aircraft with various levels of conformance capabilities, which react differently to commands by the *ATC* system and require more or less attention from the controllers.

2.7 Hazard Monitoring

Hazard monitoring is a back-up to the system consisting of ensuring separation by requiring the aircraft to conform to conflict-free flight path plans. In case of blunders by pilots or controllers, hazard detection and resolution are additional precautionary measures.

Hazards are detected by the respective collision avoidance systems of the aircraft involved or by the ground. Detecting a hazard triggers an alarm and emergency measures are then taken to resolve the hazard.

2.7.1 Alarm Threshold

Because of the discrepancy between the actual aircraft separation ΔS_{ij} and the estimates seen on-board the aircraft $\Delta \hat{S}_{ij}^{\ on-board}$ (or on the ground $\Delta \hat{S}_{ij}^{\ ground}$), detecting a hazard may not necessarily imply a dangerous encounter, and conversely an air-space violation may not be detected.

The level of the threshold of hazard detection is a parameter which determines the reliability and the relevance of the alarms. The rate of false alarm, triggered in

the case of a safe situation, can be reduced by increasing the detection threshold. Conversely, decreasing the threshold reduces the rate of missed alarm but increases the rate of false alarms.

Even in the case of a missed alarm for a given level of the detection threshold, an alarm will eventually be triggered if the aircraft continue to get closer to one another. In fact a missed alarm results in a delayed response to resolve the hazard which compromises the possibilities of reaction and increases the risk of collision.

The choice of an alarm threshold is a trade-off taking into consideration the cost of an accurate estimate of the aircraft state, the disturbance caused by an irrelevant alarm in terms of pilots and controllers workload and traffic organization, and the increased risk of collision resulting from a late reaction.

2.7.2 Hazard Resolution

Hazard resolution is a tactical problem which requires a fast response in an emergency situation. Though resolution depends on the particular geometry of the approach some general rules are applicable. The guideline is a minimum disturbance of the previous plan of air-space organization. Resolving a hazard must avoid causing another one later. Aircraft not endangered by the hazard should not be disturbed, and even only one of the affected pair should be vectored away if possible.

The means of action to locally resolve a hazard should be in order of priority: speed control, which only disturbs the timing of the assigned flight path; altitude maneuver, which conserves the two-dimensional shape of the path; and horizontal maneuver.

Resolving a hazard by simply vectoring an aircraft away is not in general an acceptable solution. The state of the aircraft after intervention of the hazard resolution process has to be compatible with the air-space organization so that the aircraft should be in a state which allows a smooth reinsertion in the flow of traffic.

After a transitory period during which the primary goal is to regain safe separations, the path conformance function re-asserts itself. Strategic considerations of flight path optimization and global air-space organization come to the foreground again once the hazard has been resolved.

2.8 *ATC* System

The evolution from a manual and tactical practice to an automated and strategic approach alters the relative importance of the function involved in the decomposition of the *ATC* system. Whereas some of the modules perform the tasks which are traditionally performed by human controllers and match current *ATC*

procedures, others become necessary in the context of automated terminal area operations.

An illusion of simplicity might result from a functional description where each module performs a logical task. Functions are inter-dependent and a global optimization of air traffic operations would require a constant feedback between the processes. These relationships are described in [32].

The prototype of a computerized Flight Path Generator is being designed to fulfill the function described above. This work is part of a long-term effort pursued at the MIT Flight Transportation Laboratory to automate the control of traffic in terminal area. The Flight Path Generator is a module to be inserted in the chain of automated processes which are being designed or built at the laboratory. Tools are available for an extensive simulation of the system.

Chapter 3

Representation of Feasible Flight Path Plans

The knowledge representation choices involved in the design of a Flight Path Generator are discussed in this chapter. An adequate formalization of aircraft motion in space is crucial since it conditions expressiveness and computational tractability of the representational framework in which aircraft trajectories are manipulated. A State Space formulation is presented. Similarly, the efficiency of path generation and the capability of replanning to accommodate operational deviations depend on the structure of flight path plans. Based on the chosen representation of trajectories, a clear semantics is designed to describe structured flight path plans.

3.1 Adequacy of a Representation

For the purpose of a Flight Path Generator, an adequate representation of aircraft motion should provide the level of detail needed by controllers to guide aircraft in terminal and final approach areas. A model of aircraft motion and space which would involve the complete rules of kinematics, geometry and mechanics, would be superfluous and hardly tractable. Besides, the values of some of the physical parameters involved might not be available. On the other hand, an oversimplification of the possible moves of aircraft would degenerate to a "toy problem" which would not be of any help to Air Traffic Controllers. Since most degrees of freedom that involve rotation of the aircraft around its center of inertia may be ignored for the purpose of Air Traffic Control, modeling an aircraft as a point in a three-dimensional space may be sufficient. Describing landing operations, that require very specific maneuvers, from a macroscopical point of view only, may prove to be satisfying too.

Numerical computations to solve kinematics, geometry or mechanics equations must be performed. However, the amount of numerical computation should be

reduced for the benefit of symbolic computation to make reasoning about motion in space and flight plans possible. Abstraction should allow reasoning in terms of aircraft maneuvers, as they are planned by controllers or performed by pilots, rather than reasoning in terms of mathematical relations.

3.2 Formulation as a State Space

The chosen representation of aircraft motion in space is state-based. Typically, the world is modeled as a sequence of states which are data structures giving snapshots of the current condition of the world at successive stages. Transitions between states are brought through the occurrence of actions which operate on states.

The State Space contains all the possible positions of the aircraft in the airspace. Some of them may not be practically acceptable. The complete specification of a State Space problem has three components:

Initial States $\mathcal{I}$ The initial state is associated with the arriving aircraft, when it is supposed to enter the area of the controller. The current position of the aircraft and the value of the kinematic parameters are data.

Goal States $\mathcal{G}$ The goal state is associated with the landing aircraft. The aircraft reaches the runway in a specified landing configuration, at the time imposed by the scheduling process. The value of the kinematic parameters may depend on the type of aircraft, weight, or on other external parameters such as airport elevation, wind, temperature. A small margin in the landing time may be acceptable.

A set of operators $\mathcal{O}$ A set of operators describes the possible actions for moving in the State Space. Available operators are the feasible aircraft maneuvers, or an absence of actual maneuver, during a time interval. The set of operators must include:

- uniform move
- accelerate
- decelerate
- climb
- descend
- turn
- turn right

- turn left.

A State Space problem is then the triple ($\mathcal{I}$, $\mathcal{O}$, $\mathcal{G}$). A solution to the problem is a finite sequence of application of operators that change the initial state into the goal. A state space can be treated as a directed graph whose nodes are states and whose arcs are operators transforming one state into another. A solution trajectory is a path from an initial node to a goal node.

3.2.1 Representation of a State

States have a uniform representation all along the trajectory. A state must contain information to describe the current situation of an aircraft so that it may provide a base for reasoning and calculation when building a trajectory. Redundant information is acceptable to speed computation.

A state is a frame composed of the following slots:

- Current time, assuming that the aircraft and the ground clocks have been synchronized.
- Position of the vehicle with reference to the ground. The coordinates correspond to a three-dimensional cartesian space in ground axis. The third component represents the altitude.
- Air-speed. Air-speed indicators measure dynamic pressure and relate to true air-speed when instrument, position, compressibility, and density corrections are applied.
- Ground-speed, module of the velocity with reference to the ground.
- Direction of flight, direction of the horizontal projection of the velocity with reference to the ground at the current point of the trajectory.

Additional parameters such as heading, bank-angle or angle of attack are not explicitly involved in the description of the trajectories. They will appear in the model of the aircraft trying to fly the generated trajectories. An over-detailed description of the current situation, though not desirable, should not sensibly affect the tractability of the method.

Air traffic controllers organize the traffic from the ground. Similarly, flight paths lead the aircraft between points of the terminal area which are defined in ground coordinates (entry points, holding fixes, runways). For these reasons, the successive positions of an aircraft which constitute a flight path are defined with reference to the ground. On the other hand the kinematics of the aircraft can not be dissociated from the aerodynamics of the flight and the motion of the air mass. Take-off and landing maneuvers or the optimization of the flight in range

or endurance impose the speed of the aircraft with reference to the air. Assuming the knowledge of the local wind, air-speed and ground-speed are deducible one from the other. The effect of wind on the definition and construction of the flight path is treated in 3.3.

Operators are the complementary part of a State Space representation of the aircraft trajectories. They represent the possible transitions between the states.

3.2.2 Representation of an Operator

In the representation described above, operators are usually viewed as opaque functional subroutines. Given a state as input, they deliver a state as output. With a uniform representation of states, operators are compatible; the output of one can be fed into another. Each operator is associated with a set of parameters which characterize the transformation. Operators contain all the necessary information to describe the motion of an aircraft between two states so that it is possible to infer the precise state of an aircraft at any intermediate time.

An absence of wind is first assumed in order to focus on the notion of operator and their representation. In these conditions, ground-speed and air-speed are identical. The common state variable is denoted v. Direction of flight, denoted d, and heading are also the same. The effect of wind is then taken into account, involving more complex equations.

The following example shows how an operator can be implemented in a "procedural" style. This operator represents a speed modification with a constant acceleration α during a time interval of length δt, and transform $State_1$ into $State_2$:

$$State_1 \xrightarrow[(parameters)]{Operator} State_2$$

$$\begin{pmatrix} t_1 \\ x_1 \\ y_1 \\ z_1 \\ v_1 \\ d_1 \end{pmatrix} \xrightarrow[(\delta t\ \alpha)]{Acceleration} \begin{pmatrix} t_2 \\ x_2 \\ y_2 \\ z_2 \\ v_2 \\ d_2 \end{pmatrix}$$

$State_1$ and the parameters of the transformation are supposed to have a known value so that it is possible to infer the value of $State_2$. The application of the operator involves the following computation:

$$\left\{\begin{array}{lcl} t_2 & \leftarrow & t_1 + \delta t \\ x_2 & \leftarrow & x_1 + \delta x \\ y_2 & \leftarrow & y_1 + \delta y \\ z_2 & \leftarrow & z_1 \\ v_2 & \leftarrow & v_1 + \delta v \\ d_2 & \leftarrow & d_1 \end{array}\right. \quad with \quad \begin{array}{lcl} \delta v & \leftarrow & \alpha\, \delta t \\ \delta l & \leftarrow & \frac{1}{2}(v_1 + v_2)\delta t \\ \delta x & \leftarrow & \delta l\, \cos d_1 \\ \delta y & \leftarrow & \delta l\, \sin d_1 \end{array}$$

Prescriptive means are provided to express how a result value can be computed from a set of input values. The variable on the left hand side of the binding arrow is bound to the value resulting from the computation of the expression on the right hand side. The necessary ordering of the bindings for a sequential execution has been relaxed. With this procedural approach, flight path plans are implicitly computed in a time-monotonous manner, forward from an initial state or backward from a final state.

However, the notion of operator may be extended to allow stating relationships in a more "declarative" way. The following operator describes a uniform move between $State_1$ and $State_2$:

$$State_1 \quad \overset{Operator}{\underset{(parameters)}{\longleftrightarrow}} \quad State_2$$

$$\begin{pmatrix} t_1 \\ x_1 \\ y_1 \\ z_1 \\ v_1 \\ d_1 \end{pmatrix} \quad \overset{Uniform\ move}{\underset{(\delta t\ \delta x\ \delta y\ \delta l)}{\longleftrightarrow}} \quad \begin{pmatrix} t_2 \\ x_2 \\ y_2 \\ z_2 \\ v_2 \\ d_2 \end{pmatrix}$$

The fact that the "uniform move" operator links these two states is expressed by the following set of relationships:

$$\left\{\begin{array}{lcl} t_2 & = & t_1 + \delta t \\ x_2 & = & x_1 + \delta x \\ y_2 & = & y_1 + \delta y \\ z_2 & = & z_1 \\ v_2 & = & v_1 \\ d_2 & = & d_1 \end{array}\right. \quad with \quad \begin{array}{lcl} \delta l & = & v_1\, \delta t \\ \delta x & = & \delta l\, \cos d_1 \\ \delta y & = & \delta l\, \sin d_1 \end{array}$$

This system of equations will be used as an example to show how the relationships characterizing an operator may be transformed into a "network of constraints" according to the method developed in chapter 5. These relationships are

declarative statements describing the logic of the problem just like mathematical relations. The expressions on both sides of the equality sign play identical roles. No commitment is made about how this knowledge will be used. An underlying system should exercise the control and determine the flow of computation.

A "declarative" approach avoids an unnecessary restriction in the role of operators which introduces some limitations on the future flows of computation and information at the early stage of knowledge representation. Operators are not restricted to a one-way functioning by a procedural formulation of their effects. They allow an arbitrary flow of information.

Some redundancy, as it appears among the parameters of the operator described above, also increases the expressive power since an operator may be applied with different perspectives. For example, a uniform move of a given duration or for a given distance can be expressed by the same operator. Duration, angle, final heading, speed, radius, angular speed, bank angle, length of the arc, coordinates of the center, intersection of the tangents at the end-points are all characteristics of a turn. A procedural approach would require to design an operator for each minimal subset of these parameters (which entirely characterizes the maneuver) and to select among them. That would involve a tedious replication of code which would become even worse to ensure an arbitrary bidirectionality. Besides all the necessary information may not be available to apply an operator though some useful partial results may be inferred. The computational model which is presented in chapter 5 circumvents all these obstacles and offers a framework for an efficient implementation of a declarative approach.

The example of a speed modification is used to show how realistic maneuvers may be described by the means of operators. In the example above of a procedural operator the acceleration parameter is a constant. For a more realistic model of aircraft motion in space, operator definition may rely on an empirical representation of aircraft performance characteristics. More complex acceleration or deceleration profiles which would depend on the specificity of each aircraft may be represented by a juxtaposition of linear phases or by a concise and computationally efficient polynomial approximation. In the first case the operator is defined as a sequence of constant acceleration operators. In the second one the polynomials become parameters of the operator. Flight test results for the $B737$ aircraft [5] provide numerical mathematical expressions characterizing aircraft performance. For an horizontal acceleration at maximum cruise power the expressions of the acceleration time and distance as function of speed and altitude are the following:

$$\begin{cases} \delta t &= (v_2 - v_1)(P_a + (v_1 + v_2)P_b) \\ \delta l &= \delta t\,(v_1 + v_2)P_c \end{cases}$$

where P_i's denote second-degree polynomial functions of the altitude. The expressions are similar for a deceleration at flight-idle power with however different coefficients for P_a and P_b. Another study based on flight test results for the $DC10$

aircraft [19] suggests that modeling acceleration as constant at various altitudes and gross-weights is an acceptable approximation.

The implementation of operators involving aircraft performance models in a pre-compiled numerical form is limited by the lack of a coherent database for various aircraft models. Constant maneuver characteristics may nevertheless be an acceptable approximation. The discrepancy with reality is limited by the fact that the durations of the maneuvers involved in the flight path plans are relatively short.

3.3 Effect of Wind

At the present time, controllers issue vector commands specifying heading, altitude, air-speed, or time at a crossing-point. These commands have to take into account the effects of the wind on the trajectories. Heading commands for instance should compensate for the drift so as to maintain a given flight direction with reference to the ground. An increasing number of commercial aircraft are capable of 4-D flight, which implies the ability to follow a specified flight path with time constraints and even to maintain a constant ground-speed as wind fluctuates. The use of a secondary surveillance radar (SSR) in the vicinity of major airports enhances the navigation capabilities (with reference to the ground) of aircraft which are not equipped with state-of-the-art avionic systems. Navigation systems make it possible for the aircraft to follow a given ground-path and maintain an average air-speed in spite of local wind variations. However a complete 4-D navigation capability is not desirable since the kinematics of the flight has to satisfy aerodynamic requirements which depend on wind. For example it would be unacceptable to require an aircraft to maintain a constant ground-speed along two successive legs joined by a turn when the wind component parallel to the direction of flight varies greatly.

3.3.1 Choice of a Model of Wind

Sophisticated wind models are available for large scale modeling but their complexity and their inaccuracy at a local scale are often obstacles to their integration in terminal area operations [22]. Good estimates of the local wind can be inferred from radar and tracker data by estimating the variation in ground-speed before and after a turn at constant air-speed or by analyzing the deformation of the trajectory from an ideal circle during a turn. The experience of the influence of wind on the trajectories of previous aircraft in a particular region of the terminal area may be used by controllers for a better planning of new flight paths. Independently from the precision of the estimate, the validity of the wind forecast degrades with time. Flight path planning is a strategic function which involves

time horizons of the order of magnitude of half an hour during which wind can change.

As a first approach to the problem of integration of wind in terminal area flight path planning, wind is modeled with a zero-order approximation as a constant within the extent of the terminal area. Wind is characterized by its speed and direction and is represented by a constant vector. The next step would be to consider its variation with altitude. Wind would be represented by a vector field of horizontal invariance with a vertical profile. These two stages of wind modeling should give reasonably good results for a very moderate expense. Beyond that, a complete grid-mapping of the air-space would involve difficulties to update the wind database and would require extensive numerical methods to integrate the effect of wind along a trajectory. The effects of a uniform wind on the various segments of a flight path are presented below. Besides the wind is assumed to be constant over the duration of a flight path segment.

3.3.2 Expression of the Ground-Speed

It is often the case that an aircraft is required to follow a flight path defined with reference to the ground at a given air-speed. The orientation of the ground-speed and the module of the air-speed are specified. Assuming the knowledge of the local wind $\vec{W}$, ground-speed $\vec{V}$ and air-speed $\vec{S}$ satisfy the following local and instantaneous relation:

$$\vec{V} = \vec{S} + \vec{W}$$

From the vectorial equality one can get a relation between the modules denoted with lower-case letters:

$$s^2 = v^2 + w^2 - 2vw\cos\theta$$

where θ denotes the angle between the direction of flight and the direction of the wind. When $s \geq w\sin\theta$ an expression of the ground-speed is the following:

$$v = w\cos\theta + \sqrt{s^2 - w^2\sin^2\theta} = w_{||} + \sqrt{s^2 - w_{\perp}^2}$$

where $w_{\perp}$ represents the cross-wind component and $w_{||}$ the component which is parallel to the direction of motion. $w_{||}$ is positive in the case of a tail-wind and negative in the case of a head-wind.

With very strong cross-wind ($s < |w_{\perp}|$) it is not possible to impose simultaneously the direction of flight and the air-speed. Such weather conditions do not allow any landing or take-off anyway. Aircraft would have to be rerouted or kept on the ground. The relation between ground-speed and air-sped which is presented above is now applied to the case of each operator.

3.3.3 Effect of Wind on a Uniform Move

When the wind is uniform and constant, the case of a uniform move is trivial. Both air and ground-speed remain constant over the time-interval during which the operator is applied. The length of the uniform-move segment seen from the ground is the product of the ground-speed and of the length of the time-interval. The direction of flight is also unchanged.

3.3.4 Effect of Wind on an Acceleration or a Deceleration

An operator corresponding to a linear change of air-speed is applied over the time-interval $[t_1\ t_2]$ between $State1$ and $State2$.

$$s \ = \ a(t - t_1) + s_1$$

$$v \ = \ w_{||} \ + \ \sqrt{(a(t - t_1) + s_1)^2 - w_\perp^2}$$

$$\delta l \ = \ \int_{State_1}^{State_2} v \ dt \ = \ w_{||}\delta t \ + \ \int_o^{\delta t} \sqrt{(at + s_1)^2 - w_\perp^2} \ dt$$

Let I denote the integral of the square root and u the radical itself. In the absence of a cross-wind component:

$$I \ = \ \int_0^{\delta t} u \ dt \ = \ \int_0^{\delta t} (at + s_1) dt \ = \ \frac{a\delta t^2}{2} + s_1$$

When there is a non-zero cross-wind component, notice the following equality:

$$(at + s_1)^2 \ - \ u^2 \ = \ w_\perp^2$$

This is the cartesian equation of an hyperbola which can be parametrized as follows:

$$\begin{cases} at + s_1 & = & \epsilon \ |w_\perp| \ \cosh \phi & (\epsilon = \pm 1) \\ u & = & |w_\perp \ \sinh \phi| \end{cases}$$

$$I \ = \ \frac{w_\perp^2}{a} \int_{\phi_1}^{\phi_2} sinh^2 \phi \ d\phi \ = \ \frac{w_\perp^2}{a} \ [\ \frac{\sinh 2\phi \ - \ 2\phi}{4} \ ]_{\phi_1}^{\phi_2}$$

where ϕ_1 and ϕ_2 are such that $\cosh \phi_1 = s_1$ and $\cosh \phi_2 = a\delta t + s_1$

3.3.5 Effect of Wind on a Turn

Consider the situation where an aircraft turns with a constant angular speed ω and maintains its altitude. The trajectory with reference to the air is supported by a circle of radius R such that $s = R\omega$, where s denotes the air-speed during the turn. The equations of the horizontal projection of the motion with reference to the ground are the following:

$$\begin{cases} x &= w_x(t-t_1) &+ R\cos(\omega t + d_1) &+ x_1 &- \frac{\omega}{|\omega|}R\sin d_1 \\ y &= w_y(t-t_1) &+ R\sin(\omega t + d_1) &+ y_1 &+ \frac{\omega}{|\omega|}R\cos d_1 \end{cases}$$

The geometric transformation between the two extreme states of a turn is the commutative product of a translation and a rotation. The trajectory is supported by an arc of cycloidal curve.

Turns are often a source of conformance error. A stochastic process might be superimposed on a determinate trajectory, especially during the turns, to account for the uncertainty of the maneuvers. Since the Flight Path Generator integrates the effect of wind in the plans, pilots should not be required to compensate for any drift. A constant angular speed and a constant altitude can be maintained by adjusting bank-angle and throttle. Simpler maneuvers should improve the conformance to the assigned flight path plans. Observing an aircraft performing a turn with a constant air-speed or constant turn-rate from the ground also allows the controllers to characterize the local wind in order to update the value of the wind which is used by the planning processes.

3.3.6 Effect of Wind on a Climb or a Descent

In steady flight, power setting is the primary control of the rate of climb or descent. The rate is proportional to the excess power. Consider the case where an aircraft flying in straight-line changes altitude with a constant climb or descent rate.

If the wind remains constant with altitude, a climb or a descent maneuver can be decomposed into two uniform moves corresponding to the vertical and horizontal projection of the motion. The horizontal ground-speed is the vectorial sum of the horizontal components of the aircraft air-speed and of the wind speed (similarly for the vertical speed if the wind happens to have a vertical component).

Let $\mathcal{S}$ denote a segment involving an altitude change, and δl the length of its projection on an horizontal plane:

$$\delta l \;=\; \int_{\mathcal{S}} v_H \; dt \;=\; \int_{\mathcal{S}} (w_{H\|} + \sqrt{s_H^2 - w_{H\perp}^2}\,)\; dt$$

where the subscript H and V respectively mean horizontal and vertical component. Using the relation: $\frac{dz}{dt} \;=\; s_V + w_V(z)$, the expression of l becomes:

$$\delta l \;=\; \int_{\mathcal{S}} (w_{H\|}(z) + \sqrt{s_H^2 - w_{H\perp}^2(z)}\,)\; \frac{dz}{s_V + w_V(z)}$$

This integral expression is tractable analytically for simple wind profiles, such as a wind of uniform direction, whose speed varies linearly with altitude (linear approximation of the profile over the range of altitude of a specific maneuver). An arbitrary profile would require numerical integration techniques.

3.4 Feasibility of the Flight Paths

3.4.1 Feasible Maneuvers

A crucial concern is the feasibility of the flight paths. A flight path has to be easily flown by the aircraft for which it is generated, given its navigation and dynamic capabilities.

A basic requirement is that all operators describe feasible pieces of a trajectory. If no maneuver is being performed an aircraft flies in straight line, horizontally, at constant speed. Thus, "uniform move" is the default operator. Otherwise an aircraft may turn, change its speed or change its altitude within the limits of its capabilities. The operators which have been implemented include turns, with constant air-speed, bank-angle, and consequently rate of turn, and acceleration or deceleration maneuvers with a constant rate implying a linearly increasing or decreasing air-speed. To satisfy these constraints the operator parameters which correspond to maneuver characteristics have to be chosen with respect to the performance of each individual aircraft or the preference of pilots.

Another requirement is a feasible sequencing of available operators. For instance the succession of some maneuvers within a short time interval may be unfeasible or require an undesirable pilot or aircraft overload. The method of flight path plan construction presented in chapter 4 ensures that such global constraints are meet.

3.4.2 Continuity of the Trajectories

In the chosen representation, a trajectory is a succession of simple geometric pieces. The method used to build the trajectories implicitly circumvents the problem of first-order continuity. Operators are differentiable functions during the application time interval, and states assure continuity between successive trajectory segments. An exception is the vertical component of velocity which is piece-wise constant but is not a state variable. Second-order continuity is not required at the macroscopic level of modeling involved. Continuity of trajectories is based on the fact that operators represent actions which perform continuous changes in the world. Operators that correspond to spontaneously triggered changes could be introduced as a practical representation of short continuous physical actions with the awareness that they are a potential source of infeasibility.

3.4.3 Concurrency of Actions

Maneuvers may be performed concurrently. For instance, a pilot may be required to descend and turn at the same time. However concurrency is limited. Only a reduced set of operators is provided and only some combinations are acceptable. "Uniform move" and "turn" or "deceleration" are essentially incompatible. Simultaneous "descent" and "decelerate" are undesirable because of the difficulty to perform both accurately in practice.

A trajectory could have been described as a tuple of three plans, executed in parallel:

1. two-dimensional geometric shape
2. vertical profile
3. speed profile

Defining a "window" as a time interval corresponding to the application of an operator during which a maneuver is performed [74] [40], windows should be adjusted in the three parallel plans in order to meet their independent goal simultaneously. In addition to independent constraints in each plan, inter-plan constraints are imposed, that are mainly negative constraints since windows in different plans may be required not to overlap. This representation is very close to the common real world perception of actions in time [73]. However the machinery involved in building parallel plans and maintaining consistency between concurrent actions is unnecessarily intricate when only little concurrency is needed [45] [61].

Concurrent actions can be represented by a new more complex operator inserted in the sequence of actions, with extensible duration to allow position adjustments of intersecting windows [44]. For instance, the operator "descent-and-turn" combines the effects of two concurrent maneuvers.

3.5 Hierarchical Description

Flight path plans are described according to various degrees of abstraction. These degrees of abstraction are organized in a hierarchy. Concretely, a plan may be viewed as a tree structure which offers an increasingly detailed description as one proceeds down the tree.

The alternative levels of description, in the order of increasingly fine granularity, are the following:

1. flight path pattern
2. flight phase

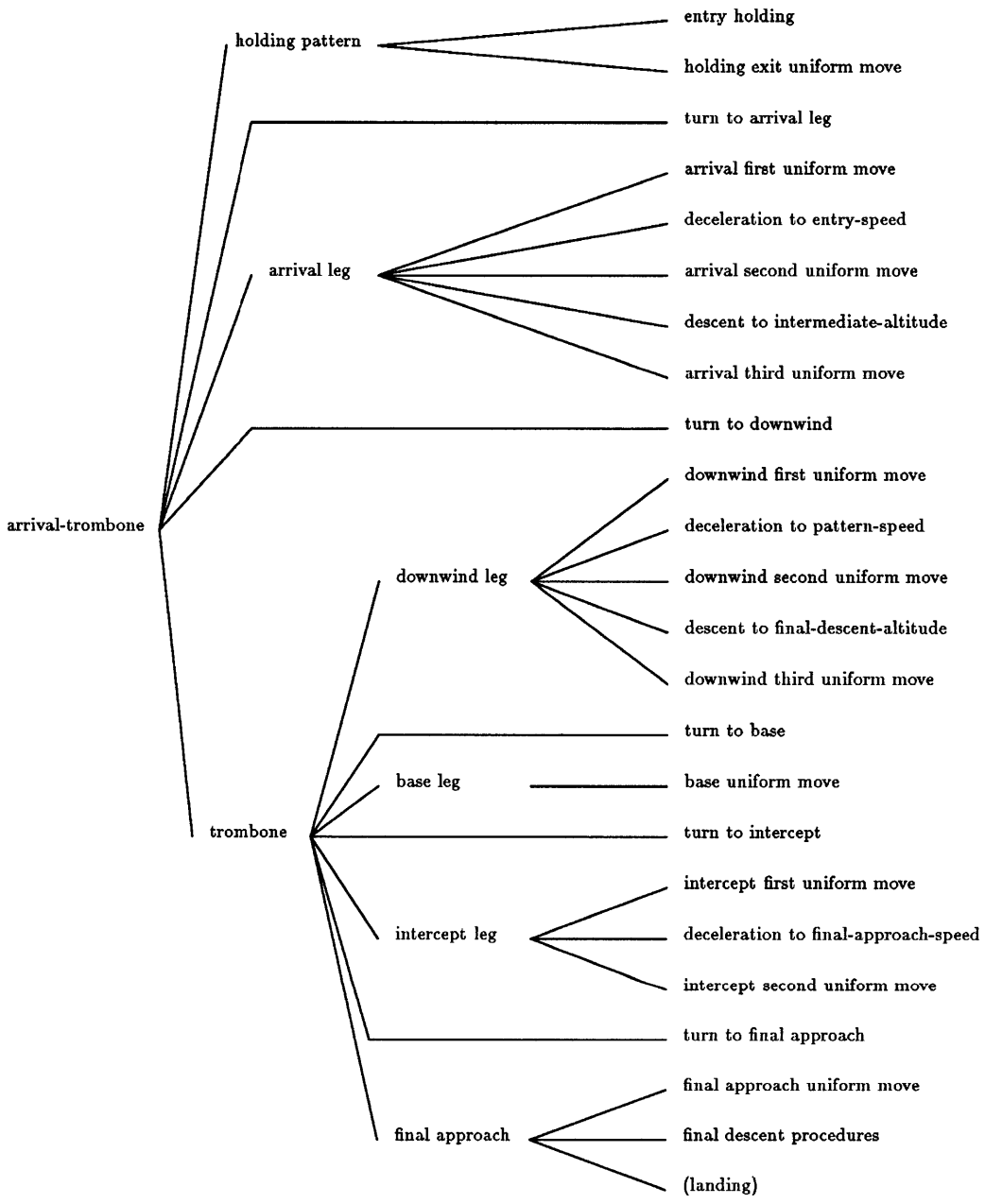

Figure 3.1: Hierarchical Representation of a Flight Path

3. segment

4. point

A flight path plan is composed of entities at a level of abstraction. Each entity may be subsequently broken into a sequence of subentities corresponding to a deeper level of abstraction. A trajectory is an instance of a flight path pattern (the notion of pattern and its use for planning purposes are described in chapter 4). A flight path pattern such as "arrival-trombone" is described as a sequence of flight phases. A flight phase may be decomposed either (such as "trombone") into a sequence of subphases, or (such as "downwind leg") directly into a sequence of segments. A segment such as "deceleration-to-pattern-speed" corresponds to the application of an operator between two states. As an elementary piece of trajectory a segment is a set of points in a four-dimensional space.

The various levels of representation are used for expressivity and modularity purposes. The underlying data structure is described in a State Space as a sequence of states and operators and consequently at the level of abstraction of the segment entity. The higher levels of abstraction are used to express properties of sequences of segments in term of flight phase or pattern and to allow a modularized manipulation of the data structure. A lower level of abstraction, in terms of points, can be used since the state of the aircraft is perfectly defined at any intermediate time between the extreme states of each segment. This lowest level allows a simulation of trajectories if necessary.

An example of this structured representation of flight paths appears in figure 3.1. The pattern "arrival-trombone" is represented in the form of a tree structure which highlights the various possible level of description. Chapter 4 gives the meaning of names on the figure which have been given to flight phases or maneuvers. The name of the pattern appears at the root. At each node the descendants correspond to subparts, at a lower level of abstraction. The description becomes incrementally detailed as one moves along the branches towards the leaves. When a leaf has been reached it is still possible to break the segment into a set of points. Though numerical functionalities to manipulate points are available, their use is reserved to particular cases when all symbolic reasoning has failed. Segments are the basic building blocks of the representational framework. The other levels of description are also widely used to abstract the reasoning and to manipulate data structures more efficiently. The construction of the flight path plans and especially the conflict detection rely upon this hierarchical representation.

3.6 Contribution of the State Space Search Formulation

A State Space representation provides a powerful conceptual framework to model actions in time and their effects. The notions of "state" and "operator" allow a simple and efficient description of aircraft motion is space with the level of detail needed by controllers. Various structural improvements and heuristics are possible from this underlying model.

Operators and sequences of operators offer a very flexible means to represent trajectories. Any path an aircraft may reasonably be required to fly can be expressed in this framework. Extensions to the presently available operators are always possible. In the presented approach, operators are declarative statements which describe elementary kinematic actions. They have a compact formulation and their application does not involve a lot of internal computation. A simple model of wind has also been incorporated in their definition to account for the effect of wind on the flight paths.

Operators fit very closely with the spatio-temporal common sense reasoning involved in the description of feasible trajectories. The basic building blocks correspond to simple geometric figures that can be manipulated symbolically. A level of abstraction has been introduced to allow reasoning in terms of aircraft maneuvers, as they are planned by controllers or performed by pilots, rather than in terms of mathematical relationships. Furthermore flight path plans can be described according to alternative levels of abstraction which are organized in a hierarchy. The highest level of abstraction is based on the use of flight path plan patterns. The next chapter describes the notion of pattern and shows how patterns can be used to plan flight paths in the terminal area.

Chapter 4

Flight Path Patterns

This chapter presents the notion of flight path pattern and shows how patterns can be used to plan aircraft trajectories in the terminal area. A simple but realistic pattern is described in detail. Its properties in terms of flexibility and applicability are analyzed. This example is used to introduce a simple model of flight path replanning in the case of schedule deviations. Among alternative patterns one is designed to accommodate missed approaches.

4.1 Notion of Pattern

Patterns offer a high level means of describing flight paths. A pattern is a predefined type of flight path with degrees of freedom. It is consequently the representative of a family of trajectories with similar characteristics and common properties.

Patterns are used as flexible and adjustable forms to build flight paths. An individual flight path which is an instance of the pattern is created by fixing some of the degrees of freedom with respect to the constraints which are imposed on the trajectory. (Examples of degrees of freedom are an angle or a distance in the geometric description of the pattern, the duration or the intensity of a maneuver.)

No notion of minimality is involved by the expression "degree of freedom". Constraints of various natures (geometric, temporal, kinematic, operational) are simultaneously imposed on the trajectory. Different subsets of the degrees of freedom can be manipulated to easily express these constraints and build flight paths from a given pattern. Degrees of freedom are deliberately redundant so that alternative perspectives of the same pattern may be considered in different situations.

In addition to a general shape which makes them widely applicable, patterns should be customized to a specific terminal area configuration. The range of variation of the degrees of freedom depends on the terminal area and on the type of the aircraft for which the flight path is created. Patterns should then be

published to allow pilots and controllers to have a mutual understanding of the possible flight paths in the terminal area.

4.2 Description of a Pattern

The flight path pattern described below has a traditional shape derived from STAR's. The aircraft diverges from an holding fix along an arrival leg to intercept a trombone-shaped figure that leads it to the runway. The aircraft stays on the side of the runway corresponding to that of its arrival. Speed and altitude are monotonically decreasing functions of time. This pattern is called "approach-trombone". Whenever the aircraft is flying straight and level at a constant speed, that segment is described as a "uniform move".

4.2.1 General Shape

The pattern is composed of the following six phases in the chronological order in which they are flown:

1. an entry holding pattern
2. an arrival leg
3. a downwind leg
4. a base leg
5. an intercept leg
6. a final approach leg

An aircraft arrives from a flight corridor at an entry point of the terminal area. Though it may have been previously metered at a remote holding fix, further delay may prove to be necessary to avoid congesting the terminal area.

Entry Holding Pattern

The aircraft may be held on its way to the runway at an holding fix close to the entry. In this case the aircraft is required to fly a planned number of "racetrack" loops at a given altitude before continuing into the terminal area. No transition between the levels of a stack is considered here. Holding loops are simply inserted at the beginning of the flight path to smoothly absorb a couple of minutes.

The aircraft leaves the holding fix and orients itself to fly along an arrival leg.

Arrival Leg

Once the aircraft has flown away from the holding fix by a sufficient distance two maneuvers are to be performed. The aircraft decelerates to an "entry-speed" and then descends to an "intermediate-altitude".

The aircraft turns to a downwind leg.

Downwind Leg

The downwind leg is a straight line parallel to the runway along which the aircraft flies downwind. A deceleration to the "pattern-speed" followed by a descent to the "final-descent-altitude" are to be performed on the downwind leg.

The aircraft turns to the base leg.

Base Leg

The base leg is orthogonal to the runway. The aircraft flies towards the runway centerline with a uniform move.

The aircraft then turns to the intercept leg.

Intercept Leg

The intercept leg makes a given angle (usually thirty degrees) with the runway. A deceleration to the "final-approach-speed" is performed.

The aircraft turns to the final approach leg.

Final Approach Leg

The final approach leg is aligned with the runway centerline. The aircraft flies level with a uniform move until it initiates the final approach procedures. It reaches the outer marker, intercepts the glide path and eventually glides down to the runway. Actual landing maneuvers are not modeled here.

The pattern "arrival-trombone" is composed of the following sequence of segments in correspondence with figure 4.1:

1. entry holding
2. holding exit uniform move
3. turn to arrival leg
4. arrival first uniform move

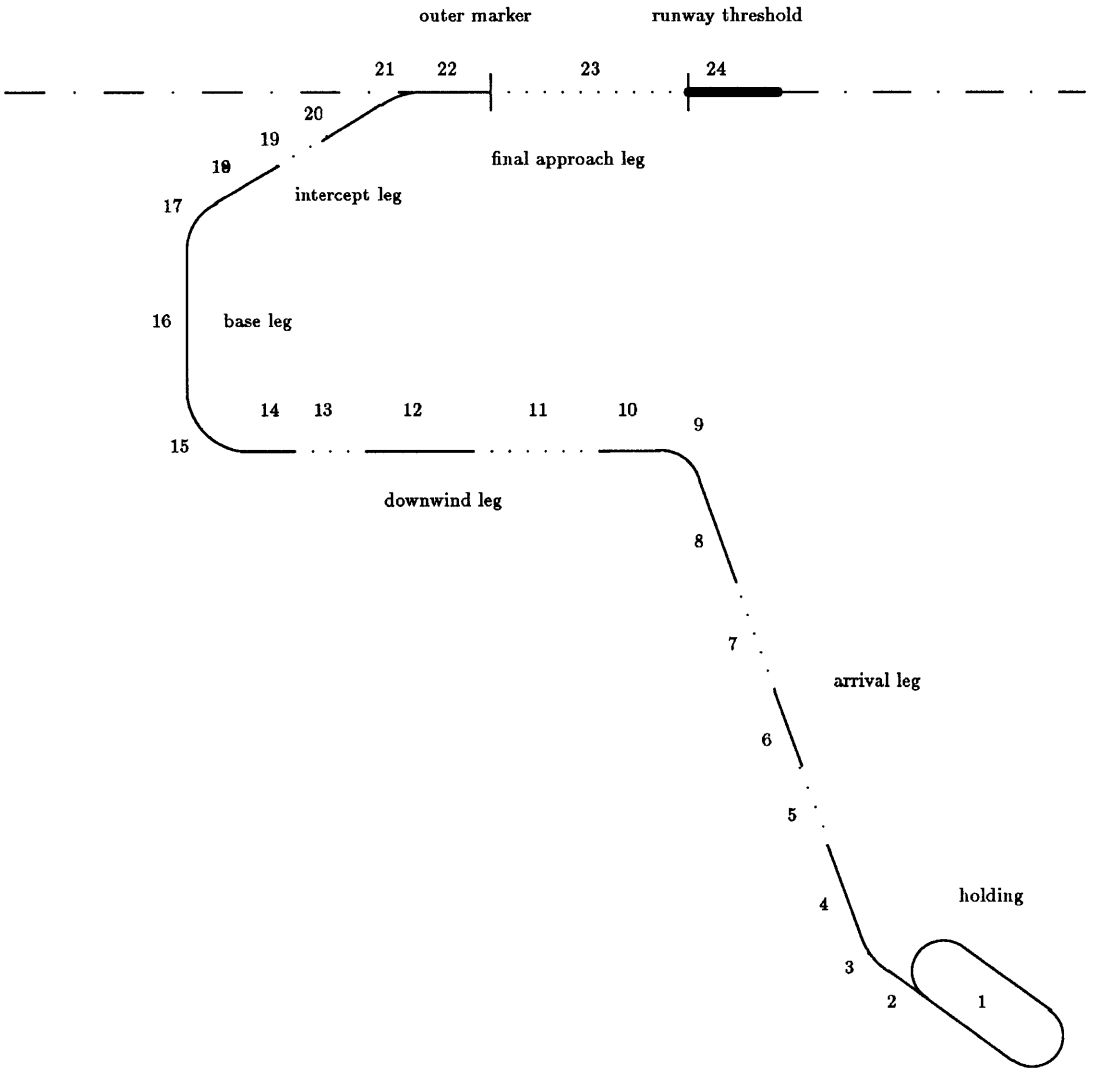

Figure 4.1: "Arrival-Trombone" Pattern

5. deceleration to entry-speed
6. arrival second uniform move
7. descent to intermediate-altitude
8. arrival third uniform move
9. turn to downwind
10. downwind first uniform move
11. deceleration to pattern-speed
12. downwind second uniform move
13. descent to final-descent-altitude
14. downwind third uniform move
15. turn to base
16. base uniform move
17. turn to intercept
18. intercept first uniform move
19. deceleration to final-approach-speed
20. intercept second uniform move
21. turn to final approach
22. final approach uniform move
23. final descent procedures
24. (landing)

Graphic conventions to draw flight path plans: the aircraft flies level with a constant speed along the path unless a dotted line indicates that a maneuver has to be performed involving changes in speed or altitude. (Dotted lines correspond to the application of operators other than "uniform move" or "turn".)

Uniform-move segments have been inserted in the pattern between the maneuvers. These segments can be stretched or shrunk but must have a minimum duration so as to ensure a minimum reaction time between successive maneuvers. That makes the paths easier to fly by reducing pilot workload in a sequence of maneuvers and can also easily express constraints such as the fact that legs must have a minimum length corresponding to a given number of radar scans for tracking purposes.

4.2.2 Degrees of Freedom

As it can be seen with the example above, patterns offer high-level descriptions of flight paths expressed in terms of kinematic and geometric properties. The choice of a pattern imposes the general shape of the path and a sequence of expected maneuvers without commitment on the choice of a specific flight path. The degrees of freedom correspond to the ability of some of the parameters involved in the description of the pattern to vary within some boundaries without compromising the characteristics of the pattern. This way various flight paths can be generated from the same flexible form. Nevertheless all the instances of the same pattern satisfy the basic constraints imposed on the flight path and exhibit the same global properties which are inherent to the pattern.

Degrees of freedom of the pattern "arrival-trombone" are the following:

- holding:
  - geometry and duration of a loop
  - number of loops (the duration of the rest of the flight to the runway is then determined)
- horizontal geometry:
  - direction of the arrival leg diverging from the holding fix
  - lateral offset of the downwind leg
  - length of the intercept leg (the length of the downwind leg and of the base leg will be uniquely determined when the other degrees of freedom are fixed)
- vertical profile:
  - intermediate-altitude
  - position of the descent along the arrival leg
  - final-descent-altitude
  - position of the descent along the downwind leg
  - descent rate for each descent phase
- speed profile:
  - entry-speed
  - pattern-speed
  - final-approach-speed
  - position of the deceleration along the arrival leg

- position of the deceleration along the downwind leg
- position of the deceleration along the intercept leg
- deceleration rate for each deceleration phase

Degrees of freedom are not meant to be independent in the sense that their values can not be fixed independently one from another. Fixing a degree of freedom by choosing a value for some parameters of the pattern restricts the choice for the other still free degrees of freedom.

A strategy to fix the degrees of freedom of the pattern and build a flight path is now presented.

4.3 Construction Strategy

Building a flight path from a given pattern consists of fixing all the degrees of freedom of the pattern. A variable is associated with each parameter characterizing a degree of freedom and the constraints imposed on the flight path are translated into relationships between the variables. The value of some of them are directly imposed, others can be inferred as consequence of the relationships which exist between the degrees of freedom or because of additional constraints. The remaining choices allow the generation of a multiplicity of operationally feasible flight paths for a given situation. Additional criteria such as the necessity of avoiding conflicts in a multiple aircraft environment, strategic considerations of terminal area organization or the preferences of a controller should be used to differentiate between the possible flight paths and "instantiate" the remaining variables. As a result of the building process, a flight path is an instance of the pattern such that all its variables have been instantiated.

The cartesian product of the sets of admissible values for each variable corresponds to the set of all the potential instances of the pattern in response to a given situation. A construction strategy is necessary to explore this set efficiently.

The use of patterns involves a trade-off between computational tractability and expressiveness. Since most variables corresponding to degrees of freedom are numerical and can take their value in a continuous interval an infinite number of flight path instances can be created from a pattern. However adjusting too many degrees of freedom or choosing among too many possible values would make the use of patterns computationally untractable. To avoid this explosion some degrees of freedom are quantized and a finite number of equidistant values are possible. On the other hand a deadlock may arise if no feasible and conflict-free flight path can be expressed by the means of the available patterns. To be able to face any situation and not be restricted by the inability of generating useful alternate paths, various flexible patterns must be known by the Flight Path Generator.

4.3.1 Choice of the Maneuver Characteristics

The maneuver characteristics involved in the construction of a flight path plan are to be adapted to the flight capabilities of each individual aircraft with respect to the Air Traffic Control procedures.

The values of the rates of deceleration, descent and turn are fixed as prerequisites to the construction of the path. The values of these kinematic parameters are either standard values for a given model or type of aircraft with similar performances or preferred values which are chosen by the pilot and must satisfy Air Traffic Control regulations.

Deceleration occurs in successive steps so as to monitor the aircraft with a speed which is appropriate to each phase of the approach. Deceleration to the entry-speed may be seen as a requirement when entering the terminal area and initiating the approach to the runway. Its value must satisfy the speed limit (usually 170 knots) imposed in the terminal area.

The pattern-speed is an intermediate speed with which the aircraft flies along most of the downwind leg, the base leg and part of the intercept leg. Its value is chosen so as to optimize fuel consumption at low altitude without compromising aircraft maneuvrability.

The final-approach-speed is the speed with which the aircraft initiates the final descent to the runway and glides down the slope fixed by the Instrumented Landing System (ILS) localizer. The final descent rate is consequently fixed. The final-approach-speed is declared by the pilot and mainly depends on the aircraft expected gross-weight at landing, the runway altitude and the atmospheric conditions.

Though the characteristics of the maneuvers such as intensity and duration are fixed early in the construction of the path, the times at which these maneuvers are performed remain adjustable.

4.3.2 Choice of an Horizontal Geometry

The choice of the flight path geometry is highly dependent on the spatial organization of the terminal area. Designing patterns in such a way that they fit in the global organization of air-space in the terminal area ensures that the generated flight paths implicitly satisfy the constraints imposed for air-space management concerns. The use of patterns offers a practical means of satisfying air-space management requirements imposed by Air Traffic Control. Figure 4.2 exhibits the degrees of freedom involved in the choice of the horizontal geometry of a flight path in the context of an example of terminal area organization.

Nominal downwind tracks are three nautical miles apart, starting at four nautical miles from the runway centerline. Aircraft with similar performance are aggregated to make the flow of traffic more uniform. The inner downwind track is reserved for slow aircraft and the outer one for fast aircraft while intermediate

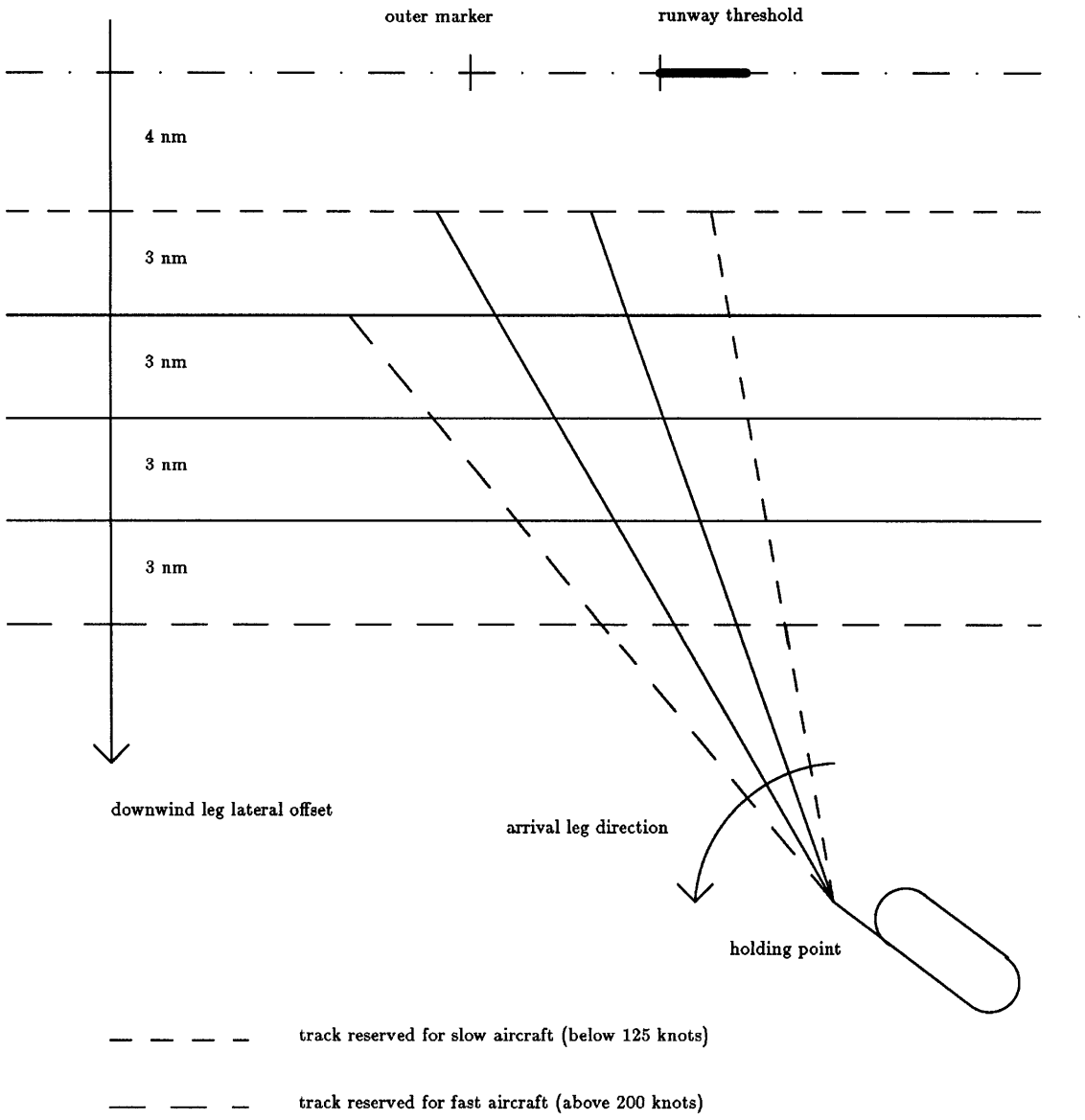

Figure 4.2: Air-space organization

tracks accept a wider mix of aircraft with the guideline of a progressive dispatching of aircraft according to their chosen pattern-speed.

Similarly the direction of the arrival leg followed by the aircraft when they diverge from the holding fixes is quantized with a ten or fifteen-degree step. From the choice of this direction depends the angles of turn to the arrival leg and to the downwind leg. A guideline to choose this direction is to minimize the sum of the angles of turn along the path so as to minimize pilot workload and avoid the "unpleasant" situations where the pilot is asked to fly back towards a place where he was a couple of minutes before. Another option which often leads to conflicting choices consists of reserving the directions which are closer to the direction orthogonal to the runway for slow aircraft so that they may intercept an appropriate downwind leg with a small lateral offset without intersecting the trajectory support of faster aircraft.

Assuming the presence of a single aircraft, most combinations of a downwind offset and of an arrival direction are almost equivalent as long as they correspond to feasible flight paths. In a multi-aircraft environment however, the potential occurrence of conflicts imposes additional constraints which differentiate these paths. The degrees of freedom may be adjusted so as to avoid conflicts. As the horizontal projection of the traffic is displayed on control screens, visually checking the separation between aircraft becomes a more intuitive task for human controllers when the aircraft are spread apart horizontally. The downwind leg offset and the arrival leg direction are degrees of freedom which can be easily manipulated to organize the traffic by separating aircraft as they diverge from a holding point or fly along downwind tracks. In fact, if separation between aircraft can be ensured by simply using the flexibility of the horizontal geometry of the path, the choice for each aircraft of a vertical profile and of the timing of the maneuvers along the path become to a certain extent independent from the rest of the traffic. This fact would make planning and plan correction much easier. For these reasons the choice of the horizontal geometry of the paths is not imposed by a strict algorithm but relies on more general guidelines which leave the door open to higher levels of decision-making. The degrees of freedom which are the more determinant for the shape of the path may be chosen according to strategic criteria of air-space organization or controllers' preference.

4.3.3 Choice of a Vertical Profile

A guideline to choose the descent profile is to postpone the descent as long as possible since flying at a lower altitude increases fuel consumption. Nevertheless there is a risk of compromising the schedule by a too late descent. Though significant saving can be gained from an optimized descent profile from cruise level, this may be disregarded in the terminal area where only a small altitude difference is involved for a short time period.

The intermediate-altitude with which the aircraft initiates a downwind leg is between the holding-altitude and the final-descent-altitude with non-strict inequalities. This way the two segments may be merged into a unique longer descent. The possible values of the intermediate-altitude are quantized every one thousand feet.

Air-space management policies may also impose restrictions on the flight levels in the terminal area. Noise abatement regions or a corridor for the outbound flow of aircraft may impose a lower or an upper-bound on the possible value of the intermediate-altitude. In a multi-aircraft environment the choice of the vertical profile is a means of ensuring vertical separation.

Unlike current practice, the altitude at which the glide slope is intercepted may be diversified to allow the simultaneous approaches of aircraft with dissimilar final-approach speeds or descent rates, thereby improving the runway throughput while maintaining a minimum time separation at the runway threshold. The same idea may be applied in the horizontal plane to the angle of interception of the runway centerline.

More complex curved trajectories may also be envisioned if the Microwave Landing System (MLS) becomes operational before its time is over.

It is important to note that not all the combinations of an arrival direction from the holding fix, a downwind lateral offset and an intermediate altitude allow building a feasible trajectory. The minimum length of a leg or the need to meet the schedule may rule out some combinations.

4.3.4 Maneuver Position Adjustment

Degrees of freedom are fixed one after the other until the pattern is fully instantiated. Assume at this stage of the construction of the flight path plan that the degrees of freedom which have been previously discussed are fixed. The maneuvers have been characterized and their duration is known but the time interval during which each maneuver is performed do not have defined end-points yet. These intervals of known length are totally ordered according to the initial sequence of maneuvers but they are still floating in time.

Adjusting the position of a maneuver phase along a leg requires characterizing the uniform-move segments of the leg. Some of these segments have been inserted in the pattern to play the role of buffers. They allow real-time tactical adjustments to compensate for the uncertainty of the real world and allow modification to accommodate changes in the schedule. Unlike in the case of aircraft maneuvers and geometrical options for which degrees of freedom had been fixed using standard or preferred values or quantized values, the duration and length of the uniform-move segments can vary continuously so as to maintain the continuity of the trajectory.

In order to safely and precisely deliver the aircraft at the runway threshold at

the scheduled landing time, the flight paths should be designed so as to enable the pilot to perform the last maneuvers in optimal conditions. Late unexpected maneuvers should be avoided. Only fine tuning at the pilot's discretion should be necessary in the late phases of the approach. For this reason the deceleration maneuver along the intercept leg is surrounded by uniform-move segments of initially fixed length; the maneuver time can be shifted for small time and speed adjustments in order to initiate the final approach leg in the best possible conditions.

In spite of inaccuracy in the maneuvers, conformance alerts may be avoided by taking advantage of the pattern flexibility. A plan may be modified by simply compressing or extending uniform-move segments. Besides if the landing schedule happens to be delayed, simply modifying the relative duration of uniform-move segments may be enough to meet the new schedule (in some cases without even altering the geometry of the path). In the case of longer delays stretching the path should allow one to provide a new feasible flight path plan. A quantitative analysis of these properties follows.

4.4 Analysis of the Pattern Flexibility

After having fixed some of the degrees of freedom by choosing a value for some parameters of the pattern according to the guidelines described above, there still remain residual degrees of freedom with uninstantiated variables. Multiple instances of the pattern can still be built from there. However previous commitments in the construction of the flight path constrain the remaining parameters. The flexibility of the pattern is restricted by a set of equalities and inequalities which are presented below.

4.4.1 General Relationships

There exists a set of general relationships which are satisfied by all paths independently of their shape. These relationships strike the balance of the construction of the path and ensure that some of the basic constraints imposed on the path are actually met. For instance these relationships express the fact that a flight path links two given positions in space in a given time. An arriving aircraft is delivered to the Flight Path Generator at an entry point of the terminal area and must reach the runway threshold at a scheduled landing time.

In a State Space representational framework, a flight path plan is represented as a sequence of operator applications linking intermediate states between two extreme states. Each operator corresponds to an elementary move and involves some discrete variations of the state variables between successive states. The sum of all the elementary variations along the path must be equal to the global difference between the extreme states:

$$\sum_{path} \Delta S_i = \Delta S_{global} = S_{final} - S_{initial}$$

Applying this equality to the components of the states leads to some equations which carry useful information to finish building the trajectories. The sum of the elementary durations of each segment along the path viewed as a sequence of segments, is equal to the time difference between the extreme points:

$$\sum_{i=1}^{23} \Delta t_i = t_{24} - t_1 \ = \ t_{landing\ schedule} - t_{delivery\ at\ a\ holding\ point}$$

Consider now the spatial counterpart of the temporal balance expressed by the previous equation. First project the trajectory into an horizontal plane, then project again on an arbitrary straight-line of this plane. The projections of the segments satisfy the following equation, where Δx_i is the oriented length of segment i along the straight-line:

$$\sum_{i=1}^{23} \Delta x_i = x_{runway\ threshold} - x_{delivery\ holding\ point}$$

This is equivalent to say that the sum of all the elementary moves parallel to an arbitrary direction for every segment along the path is equal to the corresponding distance between the extreme points. A similar projection with a different direction of projection provides a second equation which is linearly independent from the first one. The choice of orthogonal projections onto the runway centerline and onto a line orthogonal to it leads to a couple of useful complementary equations. The chosen projections reflect natural directions of the pattern "arrival-trombone" with respect to a runway. By grouping the three equations, the following system is obtained:

$$(\mathcal{E}) \quad \begin{cases} \sum_{i=1}^{23} \Delta t_i &= t_{landing\ schedule} - t_{delivery\ at\ a\ entry\ point} \\ \sum_{i=1}^{23} \Delta x_i &= x_{runway\ threshold} - x_{delivery\ entry\ point} \\ \sum_{i=1}^{23} \Delta y_i &= y_{runway\ threshold} - y_{delivery\ entry\ point} \end{cases}$$

These relationships are now applied to the specific case of the pattern "arrival-trombone". It is shown below that the system $(\mathcal{E})$ can be described as a set of linear equations.

4.4.2 A System of Linear Equations

As a consequence of the specific structure of the pattern "arrival-trombone" and of the construction strategy adopted, there exists additional relationships between the terms of the equations of $(\mathcal{E})$. At this stage of the construction of

i	Δt_i	Δx_i	Δy_i
1	$\Delta t_{holding}$	0	0
2	Δt_2	*known*	*known*
3	Δt_3	$\begin{pmatrix} \Delta x_3 \\ \Delta y_3 \end{pmatrix} = \frac{v_{HS}}{\omega} \begin{pmatrix} -\sin\theta_H & \cos\theta_H \\ \cos\theta_H & \sin\theta_H \end{pmatrix} \begin{pmatrix} \sin\omega\Delta t_3 \\ 1-\cos\omega\Delta t_3 \end{pmatrix}$	
4	Δt_4	$-v_{HS}\Delta t_4 \sin\theta$	$v_{HS}\Delta t_4 \cos\theta$
5	Δt_5	*known*	*known*
6	Δt_6	$-v_{ES}\Delta t_6 \sin\theta$	$v_{ES}\Delta t_6 \cos\theta$
7	Δt_7	*known*	*known*
8	Δt_8	$-v_{ES}\Delta t_8 \sin\theta$	$v_{ES}\Delta t_8 \cos\theta$
9	Δt_9	$\begin{pmatrix} \Delta x_9 \\ \Delta y_9 \end{pmatrix} = \frac{v_{ES}}{\omega} \begin{pmatrix} -\sin\omega\Delta t_9 \\ 1-\cos\omega\Delta t_9 \end{pmatrix}$	
10	Δt_{10}	$-v_{ES}\Delta t_{10}$	0
11	Δt_{11}	*known*	*known*
12	Δt_{12}	$-v_{PS}\Delta t_{12}$	0
13	Δt_{13}	*known*	*known*
14	Δt_{14}	$-v_{PS}\Delta t_{14}$	0
15	Δt_{15}	*known*	*known*
16	Δt_{16}	0	$v_{PS}\Delta t_{16}$
17	Δt_{17}	*known*	*known*
18	Δt_{18}	*known*	*known*
19	Δt_{19}	*known*	*known*
20	Δt_{20}	*known*	*known*
21	Δt_{21}	*known*	*known*
22	Δt_{22}	$v_{FAS}\Delta t_{22}$	0
23	Δt_{23}	*known*	*known*

Figure 4.3: Elementary Variations

the flight path, some terms have a known value. The maneuver characteristics have been fixed. Speed is known at each state. To decouple the choice of a vertical profile from the horizontal geometry of the path, assume that a value has been chosen for the intermediate-altitude among the available quantized values. The shape of the pattern imposes the direction of most legs. The uniform-move segment at the transition between the entry holding pattern and the arrival leg was introduced so that no maneuver has to be performed before the aircraft has flown away from the holding stack. Its length is fixed. The characteristics of the intercept leg are fixed for an optimal interception of the final approach leg. Other relationships are simple geometric or kinematic constraints. Figure 4.3 summarizes these relationships in a tabular form. The index i in the leftmost column reflects the ordering of the sequence of segments in correspondence with figure 4.1. The expression of the elementary spatial variations are given in term of time variations. v_{HS}, v_{ES}, v_{PS}, v_{FAS} represent respectively the holding-speed, the entry-speed, the pattern-speed and the final-approach-speed. θ_H represents the angle between the y-axis and the direction of the holding pattern. For clarity of presentation it is assumed that the pattern is built on the same side of the runway as in figure 4.1. Considering these additional relationships, the system of equations becomes:

$$(\mathcal{E}') \quad A \begin{pmatrix} \Delta t_4 \\ \Delta t_6 \\ \Delta t_8 \\ \Delta t_{10} \\ \Delta t_{12} \\ \Delta t_{14} \\ \Delta t_{16} \\ \Delta t_{22} \end{pmatrix} = \begin{pmatrix} T - t_3 - t_9 \\ X - \Delta x_3(\theta) - \Delta x_9(\theta) \\ Y - \Delta y_3(\theta) - \Delta y_9(\theta) \end{pmatrix}$$

where A is the matrix:

$$A = \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ -v_{HS}\sin\theta & -v_{ES}\sin\theta & -v_{ES}\sin\theta & -v_{ES} & -v_{PS} & -v_{PS} & 0 & v_{FAS} \\ v_{HS}\cos\theta & v_{ES}\cos\theta & v_{ES}\cos\theta & 0 & 0 & 0 & v_{PS} & 0 \end{pmatrix}$$

T, X, and Y are constants of known value. θ represents the angle between the y-axis and the direction of the arrival leg and ω the angular speed. They satisfy:

$$\theta = \frac{\pi}{2} - \omega\Delta t_9 = \theta_H - \omega\Delta t_3$$

The expression on the right hand side becomes:

$$\begin{pmatrix} T' \\ X' \\ Y' \end{pmatrix} \quad with \quad \begin{cases} T' &= T - \frac{1}{\omega}(\frac{\pi}{2} + \theta_H - 2\theta) \\ X' &= X - \frac{v_{HS}}{\omega}(\cos\theta_H - \cos\theta) + \frac{v_{ES}}{\omega}\cos\theta \\ Y' &= Y - \frac{v_{HS}}{\omega}(\sin\theta_H - \sin\theta) - \frac{v_{ES}}{\omega}(1 - \sin\theta) \end{cases}$$

Δt_3 and Δt_9 have not been treated as the other variables of the system. Some properties of the pattern have been used to combine them in a unique variable θ, and the non-linearities introduced by the presence of the trigonometric expressions have been isolated on the right hand side. These elementary durations reflect a "microscopic" description of the path at the level of the underlying pattern structure. The direction of the arrival leg can be manipulated at a higher level of abstraction to express macroscopic properties of the path more easily in the context of a structured air-space. According to the air-space organization of the terminal area presented in figure 4.2, the angle θ is quantized and only a restricted set of equidistant values is available. The direction of the arrival leg could then be fixed by a higher logic, such as the controller himself. In the following, θ is considered to be known along with T', X', Y'. $(\mathcal{E}')$ has the form of a system of linear equations in which the variables are the elementary durations of the segments along the path.

4.4.3 A Linear Program

The variables of the system $(\mathcal{E}')$ are subject to other constraints. The requirement of a minimal length or duration of the uniform-move segments may be expressed by the inequalities:

$$(\mathcal{I}) \quad \forall i \in I \ , \ \Delta t_i \geq \Delta t_{i\,min} \ \ with \ I = \{4, 6, 8, 10, 12, 14, 16, 22\}$$

These constraints on the length of uniform-move segment have been introduced in the design of the pattern to limit pilots and controllers' workload and to take into account the limitations of surveillance tracking systems.

The system of inequalities $(\mathcal{I})$ is progressively transformed using the relationships of the system of equalities $(\mathcal{E}')$. After a series of transformations the relationships between the residual adjustable parameters of the pattern take the shape of a relatively simple set of constraints of a linear program.

Δt_6 and Δt_8, just like Δt_{12} and Δt_{14}, play identical roles and can be aggregated into a unique variable:

$$\begin{cases} \Delta t_{6-8} = \Delta t_6 + \Delta t_8 \\ \Delta t_{12-14} = \Delta t_{12} + \Delta t_{14} \end{cases}$$

Tehn, $(\mathcal{E}')$ becomes a system of three linear equations with six unknown. These equations are used to express some of the variables as functions of the others. The former are then eliminated when they are substituted in the system of inequalities $(\mathcal{I})$. The choice of the variables to keep is suggested by strategic considerations. The latest phases of the flight path plan are especially constrained. They have to be built in zones where the available air-space is the most restricted with the goal to prepare good landing conditions. The rest of the plan is adjusted so as to

satisfy these constraints. For these reasons one chooses to keep the variables which appear in the latest phases of the plan (Δt_{12-14}, Δt_{16}, Δt_{22}), and to eliminate the ones which are involved in earlier and less committing phases (Δt_4, Δt_{6-8}, Δt_{10}). Besides this choice will allow one to easily express some properties of the pattern since Δt_{22}, Δt_{16}, and Δt_{12-14}, are closely related to the length of the final-approach leg, of the base leg and of the downwind leg. The expressions of Δt_4, Δt_{6-8}, and Δt_{10}, are extracted from the system ($\mathcal{E}'$) and presented in the form:

$$A_1 \begin{pmatrix} \Delta t_4 \\ \Delta t_{6-8} \\ \Delta t_{10} \end{pmatrix} = \begin{pmatrix} T' \\ X' \\ Y' \end{pmatrix} - A_2 \begin{pmatrix} \Delta t_{12-14} \\ \Delta t_{16} \\ \Delta t_{22} \end{pmatrix}$$

where A_1 and A_2 are the matrices:

$$A_1 = \begin{pmatrix} 1 & 1 & 1 \\ -v_{HS}\sin\theta & -v_{ES}\sin\theta & -v_{ES} \\ v_{HS}\cos\theta & v_{ES}\cos\theta & 0 \end{pmatrix} \quad A_2 = \begin{pmatrix} 1 & 1 & 1 \\ v_{PS} & 0 & v_{FAS} \\ 0 & v_{PS} & 0 \end{pmatrix}$$

Considering the shape of the pattern, the values of the angle θ which have a practical interest satisfy: $\theta \in]-\frac{\pi}{2}\ \frac{\pi}{2}[$. In particular, the case $\cos\theta = 0$, corresponding to the situation where the arrival leg and the downwind leg are aligned, is ruled out. The equations of ($\mathcal{E}'$) are linearly independent since:

$$Det\ A_1 = -v_{ES}(v_{HS} - v_{ES})\cos\theta \neq 0$$

The system of linear equations ($\mathcal{E}'$) is then equivalent to the following set of equalities:

$$(\mathcal{E}'') \quad \left\{ \begin{array}{lllll} \Delta t_4 = & \frac{1}{v_{HS}-v_{ES}}[& - v_{ES}T' & -X' + Y'\frac{1-\sin\theta}{\cos\theta} \\ & & + \Delta t_{12-14} & (v_{ES} - v_{PS}) \\ & & + \Delta t_{16} & (v_{ES} - v_{PS})\frac{1-\sin\theta}{\cos\theta} \\ & & + \Delta t_{22} & (v_{ES} + v_{FAS})\ ] \\ \Delta t_{6-8} = & \frac{1}{v_{HS}-v_{ES}}\frac{v_{HS}}{v_{ES}}[& v_{ES}T' & +X' - Y'\frac{v_{ES}-v_{HS}\sin\theta}{v_{HS}\cos\theta} \\ & & - \Delta t_{12-14} & (v_{ES} - v_{PS}) \\ & & - \Delta t_{16} & (v_{ES} - v_{PS})\frac{v_{ES}-v_{HS}\sin\theta}{v_{HS}\cos\theta} \\ & & - \Delta t_{22} & (v_{ES} + v_{FAS})\ ] \\ \Delta t_{10} = & \frac{1}{v_{ES}}[& & -X' - Y'\tan\theta \\ & & - \Delta t_{12-14} & v_{PS} \\ & & + \Delta t_{16} & v_{PS}\tan\theta \\ & & + \Delta t_{22} & v_{FAS}\ ] \end{array} \right.$$

Introduce the notations: $\left\{ \begin{array}{lll} T_{X'} = \frac{X'}{v_{ES}} & T_{Y'} = \frac{Y'}{v_{ES}} & \\ h = \frac{v_{HS}}{v_{ES}} & p = \frac{v_{PS}}{v_{ES}} & q = \frac{v_{FAS}}{v_{ES}} \end{array} \right.$

After substitution, $(\mathcal{I})$ becomes the following system of inequalities presented in a matrix form:

$$(\mathcal{I}') \quad A(\theta) \begin{pmatrix} \Delta t_{12-14} \\ \Delta t_{16} \\ \Delta t_{22} \end{pmatrix} \geq B(\theta)$$

where A is the matrix:

$$A = \begin{pmatrix} 1-p & (1-p)\frac{1-\sin\theta}{\cos\theta} & 1+q \\ -(1-p) & -(1-p)\frac{1-h\sin\theta}{h\cos\theta} & -(1+q) \\ -p & p\tan\theta & q \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

and B the unicolumn matrix:

$$B = \begin{pmatrix} (h-1) & \Delta t_{4\,min} & +T' + T_{X'} - T_{Y'}\frac{1-\sin\theta}{\cos\theta} \\ \frac{h-1}{h} & \Delta t_{6-8\,min} & -T' - T_{X'} + T_{Y'}\frac{1-h\sin\theta}{h\cos\theta} \\ & \Delta t_{10\,min} & +T_{X'} + T_{Y'}\tan\theta \\ & \Delta t_{12-14\,min} & \\ & \Delta t_{16\,min} & \\ & \Delta t_{22\,min} & \end{pmatrix}$$

The system of inequalities $(\mathcal{I}')$ has the form of the set of constraints of a linear program. Some similarities between the inequations and the introduction of a new parameter are now used to simplify the system $(\mathcal{I}')$.

The first inequality may be written as:

$$A_{1i} \begin{pmatrix} \Delta t_{12-14} \\ \Delta t_{16} \\ \Delta t_{22} \end{pmatrix} \geq B_1$$

where A_{1i} and B_1 denote the first rows of the matrices A and B. More interesting is the fact that the second inequality may be written as:

$$A_{1i} \begin{pmatrix} \Delta t_{12-14} \\ \Delta t_{16} \\ \Delta t_{22} \end{pmatrix} \leq B_1 + \frac{h-1}{h}\left(\frac{1-p}{\cos\theta}\Delta t_{16} + \frac{T_{Y'}}{\cos\theta} - h\Delta t_{4\,min} - \Delta t_{8\,min}\right)$$

These two relationships imply the following constraint on Δt_{16} :

$\Delta t_{16} \geq \Delta t'_{16\,min}(\theta)$ with $\Delta t'_{16\,min}(\theta) = \frac{\cos\theta}{1-p}(h\Delta t_{4\,min} + \Delta t_{6-8\,min} - \frac{T_{Y'}}{\cos\theta})$

The fifth inequality may be replaced by:

$$\Delta t_{16} \geq \max(\Delta t_{16\,min}\ ,\ \Delta t'_{16\,min}(\theta))$$

The downwind lateral offset, denoted *dwlo*, is defined as the distance between the downwind leg and the runway centerline. Since traffic flows are ordered according to the previously described air-space organization, only a small number of quantized values of the downwind lateral offset are available. *dwlo* can be simply expressed as a function of the elementary time durations:

$$dwlo \;=\; |\sum_{i=15}^{21} \Delta y_i \,| \;=\; v_{PS}\Delta t_{16} \;+\; D$$

where D is a constant of known value. Using this relationship the system of inequations becomes:

$$(\mathcal{I}'') \quad A' \begin{pmatrix} \Delta t_{12-14} \\ \Delta t_{22} \end{pmatrix} \geq B'(\theta, dwlo)$$

where A' is the matrix:

$$A' \;=\; \begin{pmatrix} 1-p & 1+q \\ -(1-p) & -(1+q) \\ -p & q \\ 1 & 0 \\ 0 & 1 \end{pmatrix}$$

and B' the unicolumn matrix:

$$B' \;=\; \begin{pmatrix} B'_1(\theta, dwlo) \\ -B'_1(\theta, dwlo) - \frac{h-1}{h}\frac{1-p}{\cos\theta}(\frac{dwlo-D}{v_{PS}} - \Delta t'_{16\,min}(\theta)) \\ \Delta t_{10\,min} + T_{X'} + T_{Y'}\tan\theta - \frac{dwlo-D}{v_{ES}}\tan\theta \\ \Delta t_{12-14\,min} \\ \Delta t_{16\,min} \\ \Delta t_{22\,min} \end{pmatrix}$$

with

$$B'_1 = (h-1)\Delta t_{4\,min} + T' + T_{X'} - T_{Y'}\frac{1-\sin\theta}{\cos\theta} - (1-p)\frac{1-\sin\theta}{\cos\theta}\frac{dwlo-D}{v_{PS}}$$

When slack variables s_i's, such that $\Delta t_i = \Delta t_{i\,min} + s_i$, are introduced the system of inequalities eventually becomes:

$$(\mathcal{I}''') \quad \begin{cases} (1-p) & s_{12-14} & + & (1+q) & s_{22} & \geq & c & H_1 \\ (1-p) & s_{12-14} & + & (1+q) & s_{22} & \leq & b & H_2 \\ -p & s_{12-14} & - & q & s_{22} & \geq & a & H_3 \\ & s_{12-14} & & & & \geq & 0 & H_4 \\ & & & & s_{22} & \geq & 0 & H_5 \end{cases}$$

where a, b, c, are functions of the downwind lateral offset $dwlo$, and of the angle θ characterizing the direction of the arrival leg. The feasible region is the intersection of half-planes H_i's. Figure 4.4 shows the boundaries of the feasible region in a general case. Figures 4.5 and 4.6 show the possible shapes of the feasible region depending on the values of the parameters.

Feasibility of the set of constraints requires $b \geq 0$ and $pb \geq (1+p)a$. If these conditions are satisfied, some values can be found to instantiate the variables corresponding to the residual degrees of freedom. In this case some instances of the pattern can then be built to provide relevant feasible flight paths.

4.4.4 Absence of an Objective Function

The relationships between the residual parameters of the pattern have been turned into the set of constraints of a linear program but no objective function is to be optimized with respect to the constraints. If a meaningful goal could be set traditional resolution methods would easily lead to the preferable instance of the pattern.

Attempts to define an objective function based on criteria such as fuel consumption and pilot workload or even to quantify the vague and intuitive notion of flexibility by combining the sensitivity to the various parameters, appeared artificial or arbitrary but always complex.

With an objective function the system would, in general, have a unique optimal solution. Basic feasible solutions of the linear program are at the vertices of the polyhedron corresponding to the feasible region. They involve the saturation of some of the constraints. Consequently even small perturbations in the constraints may rule them out.

4.4.5 Flexibility

The problem at hand consists of finding "satisfying" feasible solutions. Plans have to be flexible to allow replanning by simple alterations of the initial plan. If some of the constraints are saturated, replanning from a partially executed plan may prove to be impossible even in the case of small perturbations. The remaining degrees of freedom may have been pushed to extreme values preventing all flexibility. Replanning may then require altering later phases of the approach which had been chosen optimal. The guideline is to always preserve buffers for each degree of freedom. For this reason feasible solutions have to be chosen "far" from the boundaries limiting the feasible region.

4.4.6 Tactical Corrections

Patterns allow tactical corrections to be made to the plan to meet requirements in a changing world. Short time intervals are provided between the maneuvers

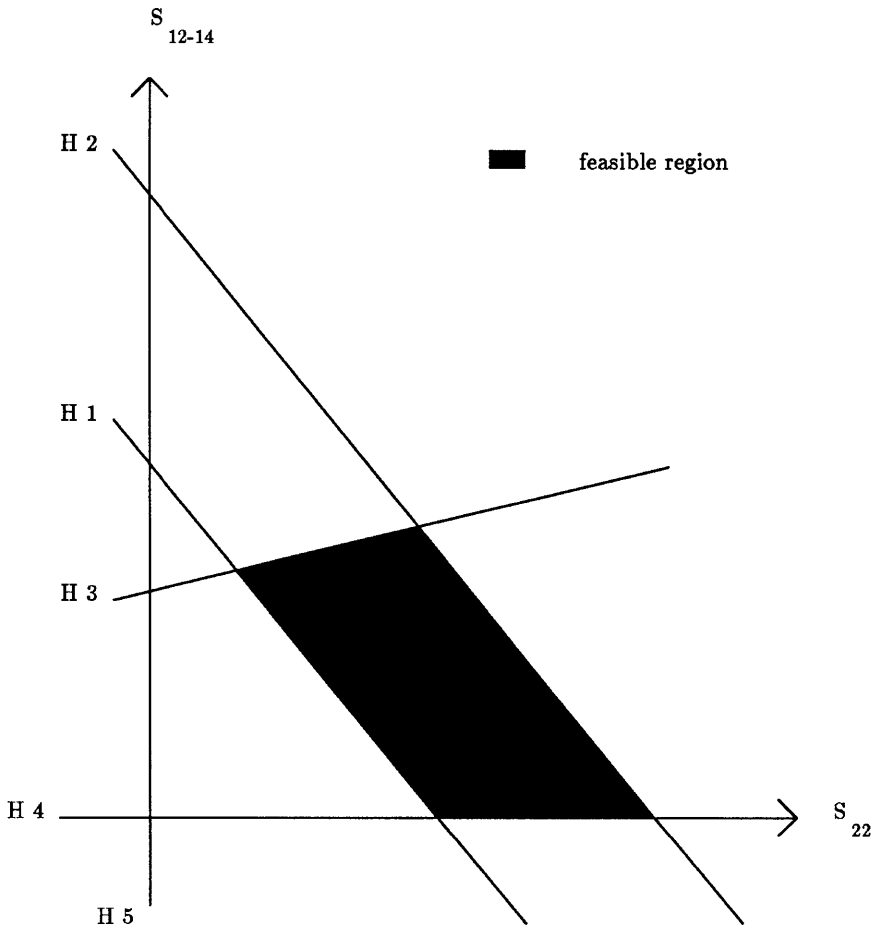

Figure 4.4: Feasible region

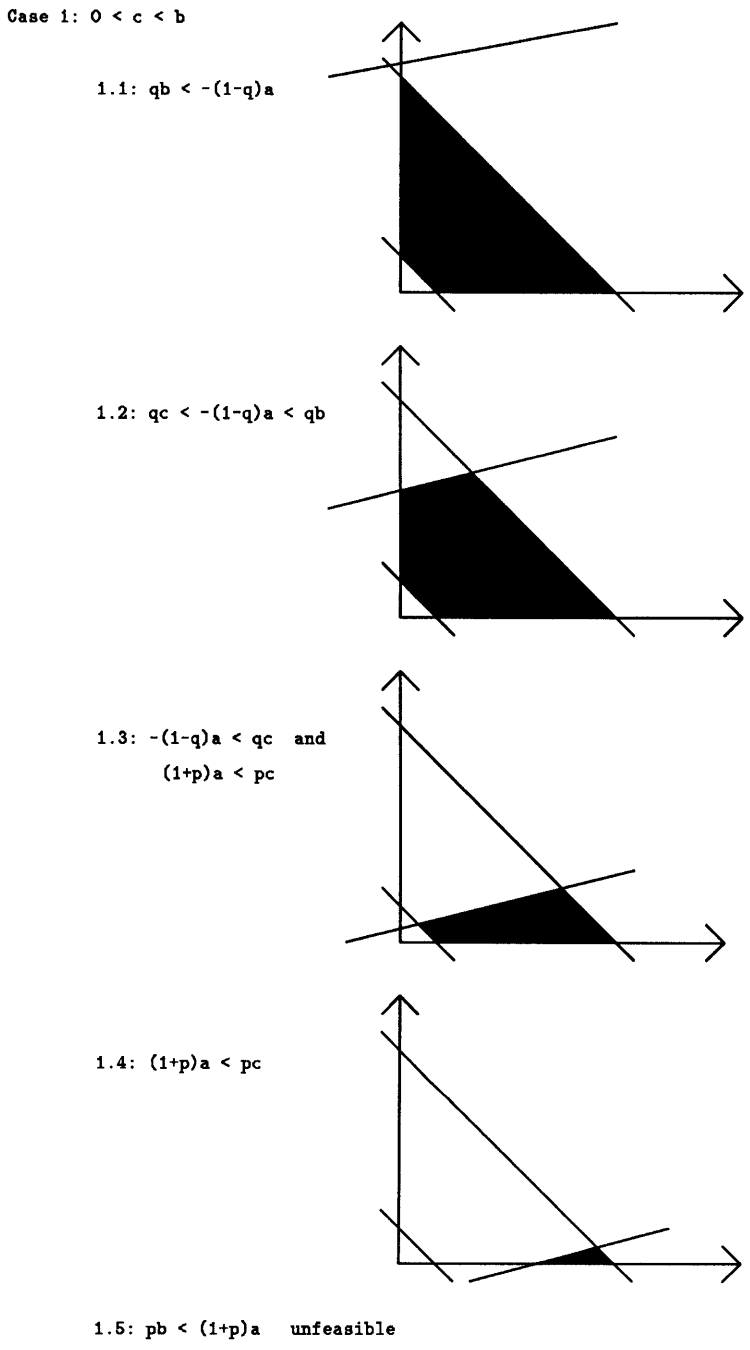

Figure 4.5: Feasibility

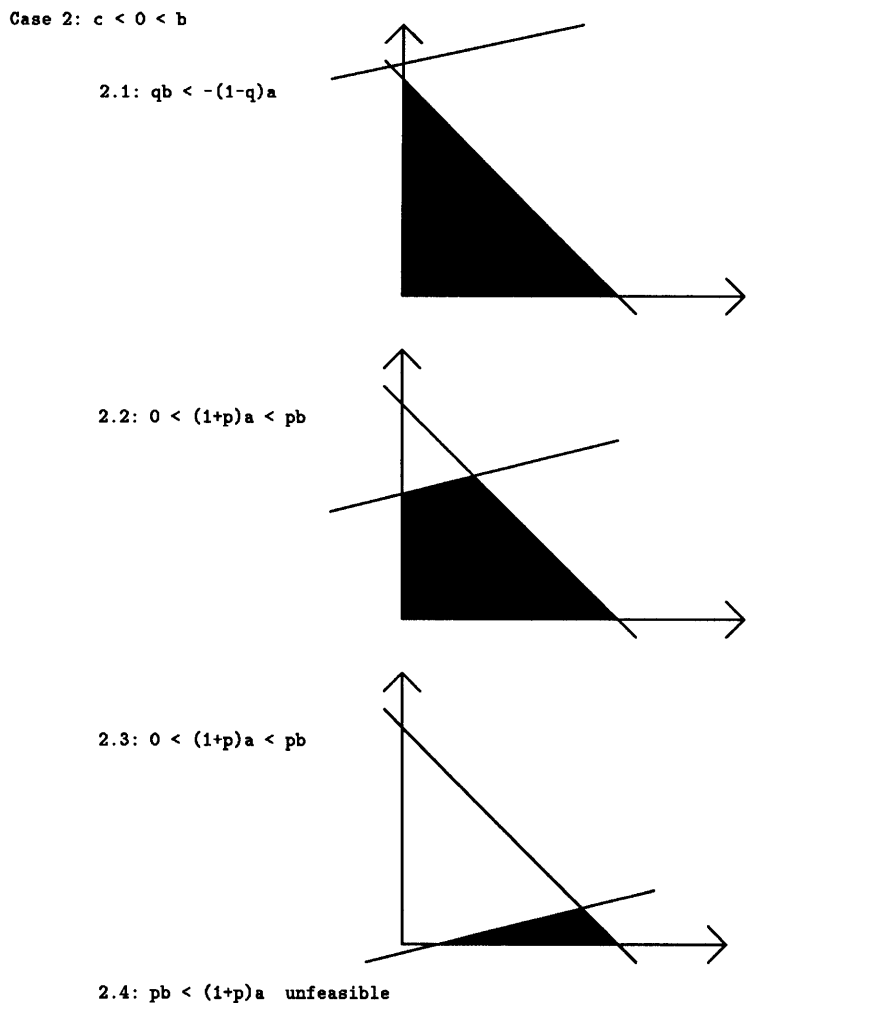

Figure 4.6: Feasibility

to absorb minor conformance leeways and avoid the accumulation of errors that would require replanning.

If a feasible solution is chosen far from the boundaries of the feasible region it will still correspond to an acceptable choice for the values of the parameters even in the case of perturbations in the position of the boundary. This capability of accommodating small perturbations in the schedule or in the execution of the plan minimizes the need for replanning.

4.4.7 Replanning

Unexpected operational deviations which involve changes in the runway schedule or conformance alerts necessitate replanning. Replanning may consist either of adapting the old plan to a new situation or of generating new flight path plans. There are some trade-offs between these two approaches. Depending on the ability and efficiency of the Flight Path Generator at providing alternative paths which depends on numerous factors such as the structure of the software system, the efficiency of the respective algorithms, and the adequacy of the hardware support, it may prove to be better to modify the old plan or to generate a new one. The extent and the consequences of the perturbation are factors which influence the efficiency and the smoothness with which the operational deviation can be accommodated.

If the aircraft has not committed its flight to the execution of the plan or if little information about the plan has been up-linked, a new plan can be generated independently from the previous one as far as the aircraft is concerned. From the ground point of view, the old plan was the result of a construction process which had taken into account the global organization of the air-space and the interaction with the other aircraft present in the terminal area. The plan belonged to a set of conflict-free paths. If the perturbations are small enough one may expect to be able to reuse most of the old plans and make them applicable again after only a few modifications and adjustments. In this case updates should be easier than the generation of entirely new conflict-free paths.

The implementation choice takes advantage of the speed with which a multiplicity of flight path plans can be generated and checked for conflict. New conflict-free flight path plans may be generated very quickly. Some of the characteristics of the old flight path plan may be kept such as the type of pattern or the values of the parameters corresponding to the main geometrical degrees of freedom. In the usual case when a change in the assigned schedule is the cause of replanning some properties of the patterns can be used to update the plans. Properties of the pattern to facilitate replanning to meet a new schedule are presented below.

4.4.8 Schedule Deviation

In reaction to an operational deviation, the runway scheduling process assigns a new landing time to an arriving aircraft. The system of equations characterizing the existing flight path plan, which is an instance of the pattern studied, happens to be modified. If the total time dedicated to following the path is increased by ΔT the coefficient b and c are both increased by ΔT . As far as the shape of the feasible region is concerned, the band limited by the two parallel straight-lines in figure 4.4 is translated to the right by the amount $\Delta s_{22} = \delta$:

$$\delta = \frac{\Delta T}{1 + \frac{v_{FAS}}{v_{ES}}} = \frac{\Delta T}{1 + p}$$

Note that in the case of a delay, $\Delta T > 0$, the area of the feasible region happens to be increased since the slope of the straight-line limiting the feasible region by above is always strictly positive. This agrees with the intuition that the more time to go, the more flexibility in choosing a path.

Trombone Effect

The main flexibility of the pattern for replanning comes from the use of a trombone-shaped path which can be extended or shortened to accommodate schedule deviations.

Let $s_i = s_i^{old\ plan} + \Delta s_i$, where s_i's are the slack variables as they appear in $\mathcal{I}'''$.

The values of Δs_{22} and Δs_{12-14} satisfying

$$\begin{cases} \Delta s_{22} + \Delta s_{12-14} = \Delta T \\ v_{FAS}\Delta s_{22} = v_{PS}\Delta s_{12-14} \end{cases}$$

generally correspond to a new feasible solution:

$$\begin{cases} \Delta s_{22} = \frac{\Delta T}{1+\frac{v_{FAS}}{v_{PS}}} = \frac{\Delta T}{1+\frac{p}{q}} \\ \Delta s_{12-14} = \frac{\Delta T}{1+\frac{v_{PS}}{v_{FAS}}} = \frac{\Delta T}{1+\frac{q}{p}} \end{cases}$$

This new feasible solution is obtained directly by a translation parallel to the straight-line of slope $\frac{p}{q}$ which limits the feasible region.

By reintroducing this solution into the system ($\mathcal{I}'''$) it appears that the first three inequalities remain unchanged, thus satisfied, since the terms of variations disappear. The last two are constraints on the amplitude of the schedule deviation which condition the applicability of the "trombone-effect" to accommodate schedule deviations. They are equivalent to the following relationship:

$$\Delta T \geq \Delta T_{min}$$

$$\Delta T_{min} \;=\; \max(\; -(1+\frac{p}{q})s_{22}^{old\ plan} \;,\; -(1+\frac{q}{p})s_{12-14}^{old\ plan} \;)$$

Advances in the schedule can be accommodated as long as they can be compensated by annihilating or reducing to a minimum length some of the uniform-move segments of the downwind leg in the old plan. Considering only the geometric and kinematic constraints which are imposed on the flight path and are inherent to the pattern, this method is applicable for all delay in the schedule. In fact the size of the terminal area or more specific spatial requirements put an upper-bound on the extent to which the pattern may be stretched.

In practice trombone-shaped figures are used by controllers in an empirical manner. They are visually very expressive and can be easily used in most cases for tactical purposes. The Flight Path Generator can take advantage of the simplicity and the wide range of applicability of the "trombone-effect" as is has been formalized above to accommodate many cases of schedule deviation.

Constant Geometry

The use of the trombone-shape of the path for replanning involves an alteration of the geometric support of the flight path which may cause conflicts with the pre-existing paths of other aircraft. A property of the flight path, which allows replanning while keeping the same geometry of the path, is now discussed.

Considering all the degrees of freedom which have already been fixed, choosing the length of the final approach leg determines the position of the base leg and consequently the length of the downwind leg. At the present stage of the flight path plan creation, keeping $s_{22} = constant$ conserves the geometrical support of the flight path for the critical part of the approach which conditions the accuracy of the arrival.

The subset of the instances of the pattern which are obtained by moving in the plan s_{22}, s_{12-14} along a vertical straight-line crossing the feasible region have an identical geometrical support. Adjustments on the flight path plan are then performed by taking advantage of the kinematic degrees of freedom of the pattern. For example short delays can be easily accommodated by decelerating earlier. On the other hand decreasing the value of s_{12-14} involves postponing the time of deceleration along the downwind leg. With the requirement of monotonous decrease of the speed and altitude of the aircraft, postponing a deceleration or a descent is a means of conserving a margin of maneuver. However, while keeping aircraft at a high altitude reduces fuel consumption maintaining them at a high speed is unnecessarily costly and complicates the control of air traffic in the terminal area.

The applicability of this property for replanning requires that the amplitude of the schedule deviation does not exceed some bounds defined so that the vertical straight-line intersects the new feasible region.

With $s_{22} = s_{22}^{old\ plan}$ the system $(\mathcal{I}''')$ implies the following relationships:

$$s_{12-14\,min} \leq s_{12-14} \leq s_{12-14\,max}$$

$$\begin{cases} s_{12-14\,min} = \max(0\ ,\ \frac{c-\Delta T-(1+p)s_{22}^{old\ plan}}{1-q}) \\ \\ s_{12-14\,max} = \min(\frac{ps_{22}^{old\ plan}-a}{q}\ ,\ \frac{b-\Delta T-(1+p)s_{22}^{old\ plan}}{1-q}) \end{cases}$$

The existence of solutions for s_{12-14} involves limitations on the schedule deviation to make it possible to build a new feasible flight path by conserving the horizontal geometry:

$$\Delta T_{min} \leq \Delta T \leq \Delta T_{max}$$

$$\begin{cases} \Delta T_{min} = c - (1+p)s_{22}^{old\ plan} - \frac{1-q}{q}(ps_{22}^{old\ plan} - a) \\ \Delta T_{max} = b - (1+p)s_{22}^{old\ plan} \end{cases}$$

The existence of a lower and an upper-bound are both consequences of the internal structure of the pattern. Replanning to meet a new schedule while keeping the same geometrical support of the flight paths is certainly attractive since it simplifies air-space organization but appears to be applicable in only restricted cases.

If before replanning, the set of conflict-free flight path plans is composed of paths which are moreover geometrically conflict-free, accommodating a schedule deviation by keeping a constant geometry of the paths has the advantage that the new paths might automatically be conflict-free. This notion of geometrically conflict-free paths is used to efficiently detect conflicts and can be used as a criterion to grade flight path plans and to choose the order in which they are built and displayed to the controller.

4.4.9 Holding

Extensible flight path plans can be built as instances of the pattern. The amount of time necessary to reach the runway can be adjusted to meet the assigned schedule by taking advantage of the degrees of freedom. However there are some limits on the amplitude of the deformation the pattern can accommodate as a consequence of its internal structure. The organization of the terminal area air-space also imposes some restrictions on the shape and the size of the paths. When a pattern includes a trombone the length of the downwind leg has to be bounded. A holding pattern is then introduced at the entry of the terminal area for lengthy delays which can not be accommodated by simple deformation of the pattern.

Assume a feasible flight path plan could be generated to meet a given schedule without making use of holding. If a holding time $T^{holding}$ is planned the amount of time allocated to fly the rest of the path from the holding point to the runway is decreased by the same amount. This is equivalent to an advance in the schedule if the construction of a feasible flight path is taken independently from the traffic. The quantities b and c characterizing the boundaries of the feasible region are decreased by $T^{holding}$ and the band limited by the two parallel straight-lines is translated to the left by the same amount.

The maximum holding time corresponds to the disappearance of the feasible region. Feasibility of the set of constraints requires $b \geq 0$ and $pb \geq (1+p)a$. Hence there exists an upper-bound on the possible value of the holding time:

$$T^{holding}_{max} = \min(b\ ,\ b - \frac{1+p}{p}a)$$

A minimum holding time may be necessary because of limitations in size of the terminal area. In other terms the pattern is not infinitely extensible. and in case of unexpected lengthy delay it may not be possible to sufficiently extend the downwind leg.

Additional holding has the advantage of keeping the arriving aircraft orderly stacked and safely spaced for a longer time instead of crowding the terminal area air-space. Final-approach controller's workload is decreased and partly transferred to the holding controller.

On the other hand providing more time to fly a pattern gives the opportunity to take advantage of wider ranges of variation for the degrees of freedom. With a larger multiplicity of possible flight paths air-space can be more efficiently used and separation criteria more easily satisfied with even additional margins of safety. Choosing a value of the holding time which is too close to the maximum value narrows the feasible region. The flexibility of the path in case of replanning happens to be restricted. Besides excessive holding may cause runway starvation.

The choice of a value for the holding time is a trade-off which involves the issues of terminal area air-space organization, controllers' workload, definition of the flight paths, flexibility for replanning, flow control.

The optional holding pattern which is incorporated at the beginning of the flight paths by the Flight Path Generator are racetrack loops which are flown level in the absence of transition commands between the levels of the stack.

The loop is composed of two half-turns which require one minute with a standard turn rate and two uniform-move segments of one or two minutes. The nominal total duration of a loop is either four or six minutes. The planned holding is composed of an entire number of loops. By combining the two kinds of loops, the holding time may be adjusted with a two-minute increment:

$$T^{holding} = 4n_1 + 6n_2 = 4 + 2n\ minutes \quad n_1, n_2, n \in N$$

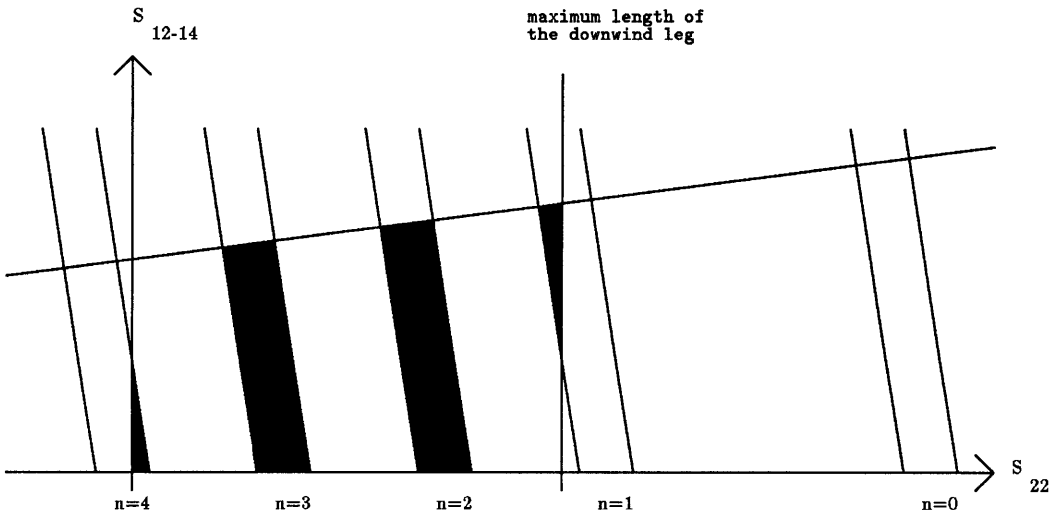

Figure 4.7: Choice of the Holding Time

To minimize the number of turn maneuvers, the proportion of six-minute loops may be maximized (for example $20 = 2 \times 4 + 2 \times 6$ and $22 = 1 \times 4 + 3 \times 6$). At the present time the exit time of a holding stack is inaccurate because of the difficulty of stable tracking when turns are frequent and the difficulty of discriminating amongst circling targets. Nevertheless it seems possible to expect the pilots to be able to fly over a navigational aid indicating the position of a holding point every four or six minutes to conform with a plan which would incorporate a few holding loops even by sacrificing the shape of the holding pattern to improve the time accuracy.

The following example illustrates the issues involved in the choice of a holding time in correspondence with figure 4.7 :

- In the absence of holding, $n = 0$, the delay can not be accommodated by simple extension of the pattern because of a restriction on the maximum length of the downwind leg.

- For $n = 1$ the area of the feasible region is very small. In case of perturbations which would involve an additional delay in the schedule the pattern could not be used.

- For $n = 4$ the pattern can not accommodate an even short advance in the schedule.

- For larger values of n corresponding to longer holding the amount of time allocated to reach the runway does not allow to build any feasible instance of the pattern.

- Cases $n = 2$ and $n = 3$ appear as acceptable compromises. Considering that delays are more probable than advances in the landing schedule the case $n = 3$ may be preferable since it makes it possible to accommodate longer delays using the flexibility of the flight path pattern.

These guidelines should allow the Flight Path Generator to efficiently include a holding pattern as a time buffer at the beginning of the flight path plans. Various holding time policies may be envisioned. The algorithm which has been implemented consists of choosing the longest holding time such that the uniform-move segments of the downwind leg and of the final-approach leg have a minimum duration. The latter condition ensures that limited advances in the schedule can be accommodated. This way holding loops are incrementally introduced in the plan to smoothly absorb a large part of the delay without compromising replanning.

4.5 Alternative Patterns

Alternative patterns are made available to diversify the choice of flight paths which can be created by the Flight Path Generator and offered to the controllers. This choice is certainly not exhaustive. On the contrary other patterns should be designed and the existing ones should be refined to meet the needs of the controllers in all situation.

Overhead-Trombone

This pattern differs from current practice by the fact that aircraft approach the runway on the side opposite to that of their arrival in the terminal area. Aircraft converge and descend to the axis, defined as a vertical line from the touchdown end of the runway, and pass overhead. They then diverge to the other side of the runway to follow a trombone-shaped figure to reach the outer marker like in the previous case.

The pattern "overhead-trombone" is composed of the following sequence of segments in correspondence with figure 4.8:

1. entry holding

2. holding exit uniform move

3. turn to convergent arrival ray

4. convergent arrival first uniform move

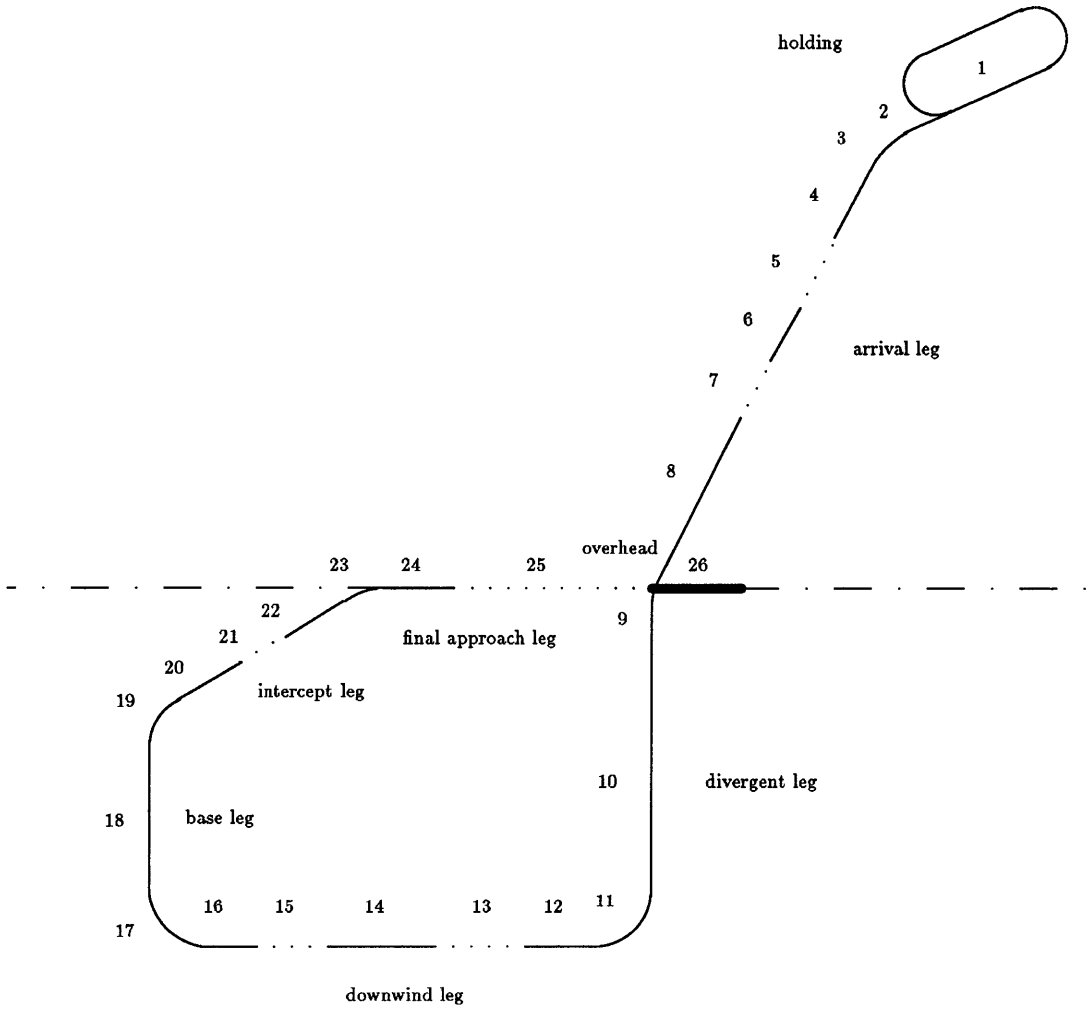

Figure 4.8: "Overhead-Trombone" Pattern

5. deceleration to entry-speed
6. convergent arrival second uniform move
7. descent to intermediate-overhead-altitude
8. approach third uniform move
9. overhead turn to divergent ray
10. divergent uniform-move
11. turn to downwind
12. downwind first uniform move
13. deceleration to pattern-speed
14. downwind second uniform move
15. descent to final-descent-altitude
16. downwind third uniform move
17. turn to base
18. base uniform move
19. turn to intercept
20. intercept first uniform move
21. deceleration to final-approach-speed
22. intercept second uniform move
23. turn to final approach
24. final approach uniform move
25. final descent procedures
26. (landing)

Aircraft are assigned an altitude at which they must cross the axis and fly overhead every one thousand feet above three thousand feet. Requiring several aircraft to converge to the same axis should not involve any bottle neck in the flow of traffic since vertical separation can be used. Aircraft have to fly level before and after the overhead turn so that ensuring separation criteria in this region of dense

radial traffic sums up to assigning different levels to consecutive aircraft. The axis is positioned over the runway but could be moved with respect to air-space organization constraints such as the geographical configuration of the terminal area.

Unlike with the pattern "arrival-trombone" the direction of the arrival leg is now forced to converge to the axis but the determination of the direction of the divergent ray introduces a new degree of freedom. The point of intersection of the downwind leg and its length depend on the value of this parameter which is quantized. It should be chosen such that neither the overhead turn nor the turn to the downwind leg involve too large angles, say not more than a quarter or five eighth of a turn.

The patterns "arrival-trombone" and "overhead-trombone" are otherwise identical (or say symmetrical with respect to the runway centerline) from the downwind leg phase till landing. They can be used in a complementary manner for air-space management. Using both alternatively allows controllers to dispatch the aircraft on both sides of the runway to balance air-space occupancy.

Arrival Direct to Base Leg

When the traffic is not congested, to insert a new aircraft in the arrival flow in reaction to an operational deviation in the schedule, or in case of emergency, the pattern "arrival-direct-to-base" provides flight paths to reach the runway quickly. The downwind leg is short-cut and aircraft turn directly to base leg. The maneuvers which occurred along the downwind leg are performed during the neighboring phases. The descents to the intermediate-altitude and to the final-altitude are merged into one descent maneuver along the arrival leg. The deceleration to the pattern-speed is postponed on the base leg.

The pattern "arrival direct to base" is composed of the following sequence of segments in correspondence with figure 4.9:

1. entry holding
2. holding exit uniform move
3. turn to arrival leg
4. arrival first uniform move
5. deceleration to entry-speed
6. arrival second uniform move
7. descent to final-descent-altitude
8. arrival third uniform move

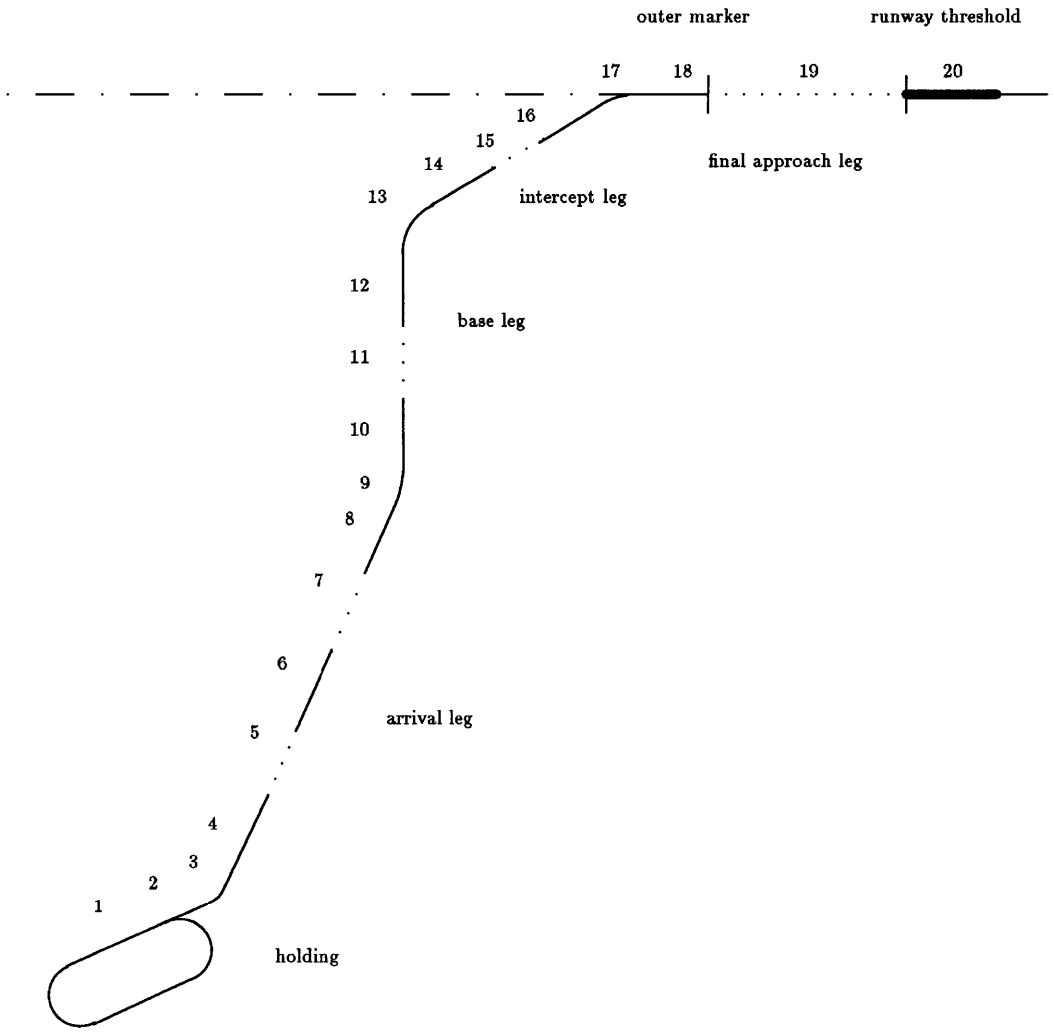

Figure 4.9: "Arrival-Direct-to-Base" Pattern

9. turn to base
10. base first uniform move
11. deceleration to pattern-speed
12. base second uniform-move
13. turn to intercept
14. intercept first uniform move
15. deceleration to final-approach-speed
16. intercept second uniform move
17. turn to final approach
18. final approach uniform move
19. final descent procedures
20. (landing)

This pattern allows the Flight Path Generator to build flight paths which are faster than with the previous two patterns in which the downwind leg may in some cases appear useless. Speed is gained however by loss of flexibility. Fewer geometrical degrees of freedom are available. The length of the final approach leg is imposed by the length and the direction of the arrival leg. The absence of a property comparable to the trombone-effect restrains the range of landing time for which the pattern is applicable Accommodating deviations in the schedule is made difficult by this lack of extensibility.

4.5.1 Missed Approach

After declaring a missed-approach the pilot needs some time to recover safely. He must perform some emergency maneuvers to gain some mechanical energy by climbing and accelerating and to reach a stabilized state. At the end of this transitory period of recovery the aircraft can fly level at two or three thousand feet with a speed which allows it to follow a low altitude flight path in the terminal area. According to missed-approach procedures the aircraft leaves the vicinity of the runway and orients itself towards a holding point where it may be stacked if the newly assigned landing schedule involves long delay. The holding phase may be short-circuited and it is possible to directly intercept a landing pattern. The aircraft is then inserted into the arrival flow of traffic along the downwind leg of a trombone which leads to the runway similarly to the previous patterns.

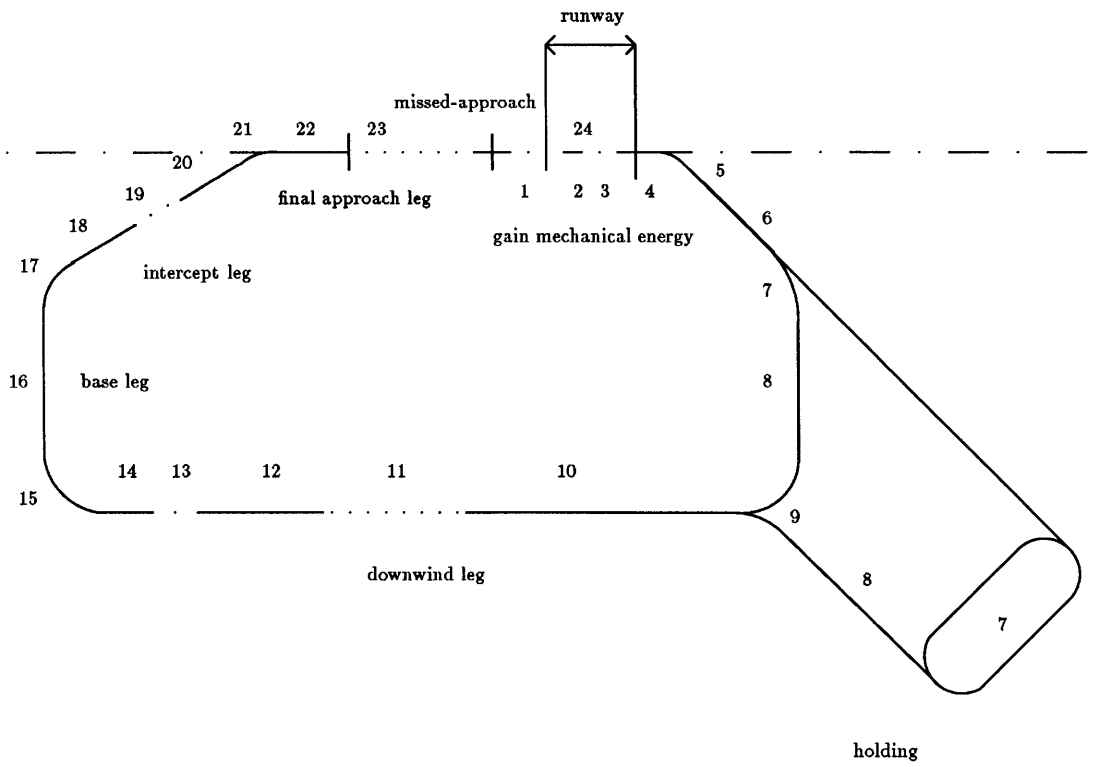

Figure 4.10: "Missed-Approach" Pattern

The pattern "missed-approach" is composed of the following sequence of segments in correspondence with figure 4.10:

1. climb to recovery altitude
2. first over runway uniform-move
3. accelerate to pattern-speed
4. second over runway uniform-move
5. turn to holding
6. first uniform move to holding
7. holding (or turn to avoid holding)
8. uniform move from holding (or uniform move to avoid holding)
9. turn to downwind
10. downwind first uniform move
11. deceleration to pattern-speed
12. downwind second uniform move
13. descent to final-descent-altitude
14. downwind third uniform move
15. turn to base
16. base uniform move
17. turn to intercept
18. intercept first uniform move
19. deceleration to final-approach-speed
20. intercept second uniform move
21. turn to final approach
22. final approach uniform move
23. final descent procedures
24. (landing)

In the case of a sudden change in visibility or runway icing condition, several successive aircraft may be declaring a missed-approach. They are then stacked at a holding point awaiting improvement in weather condition. Flight paths can be provided for these aircraft to meet a new schedule once landing becomes possible for them again. At that time they may have to be directed to another runway. Patterns have to be defined with respect to the terminal area organization to allow the aircraft to leave the holding point and to intercept a known landing pattern corresponding to the new runway. By the usual techniques the Flight Path Generator would adjust the parameters of the pattern to create a conflict-free feasible flight path which satisfies all the procedural requirements. No such pattern has been designed in this report however since it would be too dependent on the real configuration of a specific terminal area, but it would be easily accomplished.

The maneuvers which are performed just after a missed-approach declaration may not correspond exactly to what the pilot would do in an emergency situation. The purpose is to model the transitory steps of recovery to build a coherent plan assuming that the pilot will manage to climb up to a predefined altitude and accelerate according to the standard procedures. The rest of the path for which conformance monitoring is possible is then built from the state resulting from these maneuvers.

4.6 Advantages of Patterns

Patterns are described in terms of flight phases, maneuvers or vector commands. They are easily understandable by controllers who may take part in their construction and pilots who have to conform with the plans.

Patterns are aimed to be interchangeable and to evolve so that the system can be adapted to new procedures or to controllers' choices. They can also be tuned to the specific case of each terminal area. Pattern descriptions can be considered as data which are implemented independently and may be altered at very low cost and time to satisfy new requirements.

Patterns offer a practical means of satisfying air-space management requirements. They have to be designed in such a way that they reflect the spatial organization of the terminal area. The generated flight paths consequently satisfy the constraints imposed for strategic concerns of air-space management.

Pattern flexibility facilitates tactical corrections and provide a framework for replanning. Predefined degrees of freedom can be adjusted to adapt the flight paths to a changing environment. Schedule deviations can be accommodated with minimal disruption of the rest of the traffic.

Patterns provide a high-level description of aircraft trajectories. Flight paths can be manipulated analytically and symbolically. The conflict detection algorithms used by the Flight Path Generator take advantage of these complementary approaches.

Chapter 5

Generation of Flight Paths by Constraint Propagation

The generation of terminal area flight path plans is modeled as a generate-and-test heuristic. It relies on the fast generation of multiple operationally feasible flight paths which are subsequently tested for conflict. The definition and the generation process of the paths is based on the notion of constraint. This chapter formalizes the notion of constraint, describes the constraint propagation model of computation, and presents a dataflow implementation of the generation process. The dataflow and constraint propagation computational models share some fundamental properties which are exploited to efficiently construct flight paths. Parallelism, asynchrony, non-determinacy, and similarity between the structures of the underlying graphs, are some of the attractive common characteristics. Non-determinacy is introduced and controlled at the software level by some adaptations of the programming language *Id*.

5.1 Thinking in Terms of Constraints

The definition and the construction process of the flight path plans rely on the notion of constraint. A flight path plan is defined in terms of its properties which are stated as operational constraints on its definition. As more constraints are added, the description of the path becomes more and more specific. Viewed as a search strategy, adding constraints progressively reduces the size of the search space to be considered by ruling out those parts that do not satisfy all the constraints so that the search eventually converges to a solution. The effects of each constraint are propagated to infer the consequences of the interactions with the rest of the plan and to detect potential inconsistencies. At any intermediate stage of the construction an uncompleted plan satisfies all the existing constraints with a minimum commitment. Further characterization of the plan is only restricted by the previously imposed constraints and their implications.

The notion of flight path pattern has been introduced to master the infinity of trajectories which can be built in a four-dimensional space. As described in chapter 4, a pattern is a predefined type of flight path which can be used as a flexible form to build flight paths with common properties. Patterns are themselves the result of an accumulation of operational constraints with the purpose of defining suitable aircraft trajectories for terminal area operations. Their definitions are based primarily on controllers' and pilots' experience. They also reflect some properties of terminal area organization for a strategic planning of air-space activity. A flight path pattern is a partial trajectory plan in the sense that it represents a family of envisionable paths with similar characteristics rather than a definite one. Some degrees of freedom can be adjusted to generate a variety of paths adapted to each particular situation . For instance the initial conditions from which the plan is to be built and the final conditions to which the plan is supposed to lead the aircraft are stated as additional constraints on the path. Similarly, the values of the other parameters of the pattern are incrementally specified until all the degrees of freedom of the pattern are fixed. A flight path plan is then entirely specified as the result of an accumulation of constraints and of the propagation of their effects.

5.2 A Computational Model

The constraint paradigm is a computational model which allows the description of a problem in terms of constraints and provides a mechanism to handle them automatically. The principle of its usage to build flight path plans is the following. A constraint is a declarative statement of a relationship between parameters involved in the definition of a flight path pattern. A network of constraints is built as a description of a pattern, which embeds the relationship between its degrees of freedom. Propagation of the effects of the constraints is carried out automatically by an underlying mechanism. Information is propagated through the network in the form of values for the parameters. Deduction is performed locally and the results of the local computations are in turn propagated to contribute to the progressive elaboration of a flight path plan. The consistency of the network of constraints is ensured by detecting the inconsistencies in the structure of the flight path plan.

5.2.1 Declarativeness

An algorithm can be viewed as consisting of two disjunct components, the logic and the control [60]. The logic is the statement of what problem is to be solved. It expresses the knowledge that can be used in a declarative or non-procedural form. The control is the statement of how it can be solved. It specifies the resolution strategy. Traditionally, prescriptive means are provided in imperative

programming languages for the programmer to express how a result value should be computed from a set of input values. The programmer is responsible for the control, i.e. is required to impose in advance the explicit sequencing of operations and to organize the flow of information. For example, the instruction " $y \leftarrow x+1$ " indicates that at the stage of the program execution where this instruction is encountered, an addition operation should be performed to compute the value of y from the value of x. The flow of information is unilaterally imposed as indicated by the direction of the arrow. Besides, if the sequencing of operations does not ensure that the value of x is known when the instruction is executed, a run-time error is likely to occur.

The ideal of a declarative language is that the programmer only has to specify the logic of a program, while the programming system should exercise the control by determining the flow and the sequence of computation. A program in a declarative form consists of statements of relationships among symbolically named quantities which are to be satisfied. Physical systems are usually specified as sets of constraints among several variables. A constraint is a declarative statement expressing a structural relationship between some parameters of the system. In the case of a flight path which can be described by a set of mathematical equations, the constraints express kinematic and geometric relationships between parameters of the path. The way each relation is used and the order in which they are used is indifferent from the nature of the path. Expressing the path descriptions in a declarative language in terms of constraints has no prior prejudice as to the direction of the flow of computation.

5.2.2 Electrical Device Analogy

By analogy with the modeling of electrical devices, a constraint is represented by a black box which communicates with the outside world through pins. Each pin corresponds to a variable or a constant. For instance the relation " $y = x + 1$ " may be represented by a adder as depicted in figure 5.1 The pin corresponding to the first operand is associated with the variable x, the one corresponding to the second operand with the constant 1, and the one corresponding to the sum with the variable y.

A device has the power to enforce a relation between the values at its pins. Depending on the information available at the pins, the enforcement of the constraint may have different effects. If the value, say 2, is present at the pin x, the operation $y \leftarrow 2 + 1$ is performed and the value 3 is imposed at the pin y. Symmetrically, if the value 3 is present at the pin y, the operation $x \leftarrow 3 - 1$ is performed and the value 2 is imposed at the pin x. If neither x nor y have a value, no operation is performed. The simultaneous imposition of values which do not satisfy the relation is an inconsistency.

An adder describes a relation between three variables:

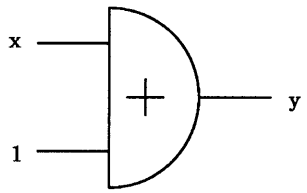

Figure 5.1: Adder

$$operand_1 + operand_2 = sum$$

and enforces the relation by performing any one of the following operations:

$$\begin{cases} sum \leftarrow operand_1 + operand_2 \\ operand_2 \leftarrow sum - operand_1 \\ operand_1 \leftarrow sum - operand_2 \end{cases}$$

The device is not restricted to a unidirectional behavior in the sense that it is not biased towards a static flow of computation. No preferential direction of computation is predefined and the behavior of the device is chosen dynamically according to the external conditions. An operation is triggered when enough information is available at the pins.

5.2.3 A Network of Constraints

The set of constraints involved in the description of a flight path plan may be organized in a network. Each constraint is represented by a physical device, and by analogy with electrical circuits the devices may be connected by wires. The pins of the devices become the nodes of the network. A wire connecting two pins expresses the fact that the values at the pins must be equal, either by identity or by equality of the corresponding variables. When a variable appears in several relations, the different instances are recognized as referring to the same parameter and the corresponding pins are connected by a wire to express this identity. An equality relationship between two variables is similarly represented by a connection. Such a network structure highlights the dependencies which exist between the constraints. Though a constraint network may be described by many set of equations, there exists exactly one network for any given formulation of a set of equations. Two equivalent equation sets are not necessarily represented by the same network since the transformation between two logically equivalent systems of equations may involve arbitrary changes in the structure of the equations.

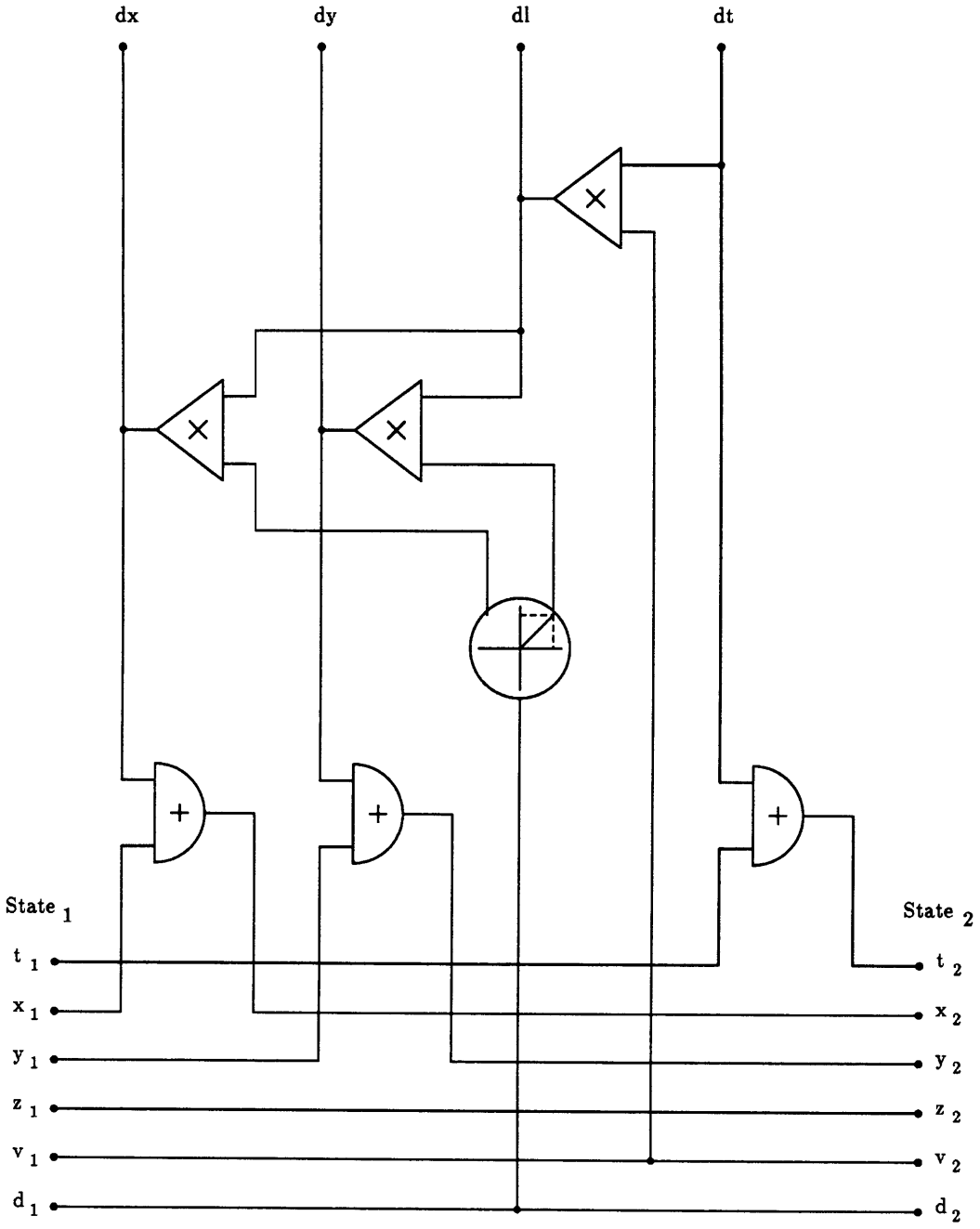

Figure 5.2: Network Representing a "Uniform-Move" Operator

The network of figure 5.2 represents a "uniform-move" operator. The operator links $State_1$ and $State_2$ and is characterized by a set of parameters. The mathematical relationships of the following system of equations are embedded in the network:

$$\begin{cases} t_2 = t_1 + \delta t \\ x_2 = x_1 + \delta x \\ y_2 = y_1 + \delta y \\ z_2 = z_1 \\ v_2 = v_1 \\ d_2 = d_1 \end{cases} \quad with \quad \begin{aligned} \delta l &= v_1\, \delta t \\ \delta x &= \delta l \cos d_1 \\ \delta y &= \delta l \sin d_1 \end{aligned}$$

The pins corresponding to the variables of altitude, speed and direction are directly connected by a wire. Adders, multipliers, and a trigonometry device (ensuring the relationship between an angle and its sine and cosine), are used to express the relationships between the state variables and the parameters of the operator.

5.2.4 Local Inference and Value Propagation

Each flight path pattern is represented by a network of constraints. Local inference and value propagation are the mechanisms used to effectively build flight path plans from this network structure. Local inference is performed by each device. A device uses the information which are locally available to infer new information. The new piece of information is propagated in the rest of the network to the devices which are susceptible to use it. A path is incrementally constructed by the combination of these two mechanisms.

A device communicates with the network by its pins. As soon as enough values are known and present at its pins, the device is activated. Depending on the combination of the variables which have a known value, different operations may be performed. The device internally computes new values and exhibits the result at the designated pins. The new values flow along the wires and are automatically propagated to other parts of the network where other devices may use them in their inference process. Devices are black boxes which enforce a constraint between the variables at their pins. Each device operates independently from the circuit to which it is connected. Similarly the propagation of values along the wires is asynchronous.

5.2.5 Restrictions of Locality

The mechanisms of local inference and value propagation provide a simple method to solve systems of equations which are represented by a network of constraints. While arbitrarily complex systems can be represented in this framework, resolution is inherently limited by the locality of the inference process.

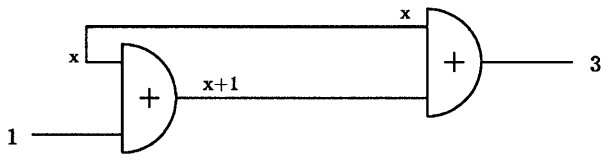

Figure 5.3: Deadlock

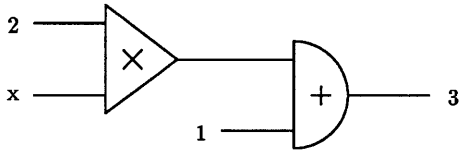

Figure 5.4: Possible Resolution

Figure 5.3 provides an example of a circuit which is deadlocked because of the limitations of the resolution mechanism. No inference is possible though all the necessary information to complete the resolution is available. None of the devices has locally enough information to be able to operate and deduce new information. If the devices could cooperate they would come to a conclusion. However, local inference only manipulates values and ignores the full power of algebraic transforms. A more global analysis is necessary. The circuit represents the equation: " $x + (x + 1) = 3$ ". The resolution mechanism does not know about the associativity of addition to deduce " $(x + x) + 1 = 3$ " and can not transform a sum of equal terms into a product to obtain the equivalent equation " $2x + 1 = 3$ ". The circuit of figure 5.4 represents an equation which is equivalent to the one leading to the deadlocked circuit of figure 5.3, but this time it is possible to infer that $x = 1$. The loop in the network has been replaced by a tree structure.

The equation " $xy + 3y = z$ " is represented by the circuit of figure 5.5.a. The value of the variable y is used by both multipliers. Unfortunately the multipliers operate locally and can not see that their pins are connected. This additional piece of information is lost and resolution can not proceed. By recognizing that the variable y appears in both products, the law of distributivity of multiplication over addition is an algebraic tool which may be used to transform the network into a tractable form [69]. The equation becomes" $(x + 3)y = z$ ". The corresponding network appears in figure 5.5.b. As a result of the transformation, a cycle has been

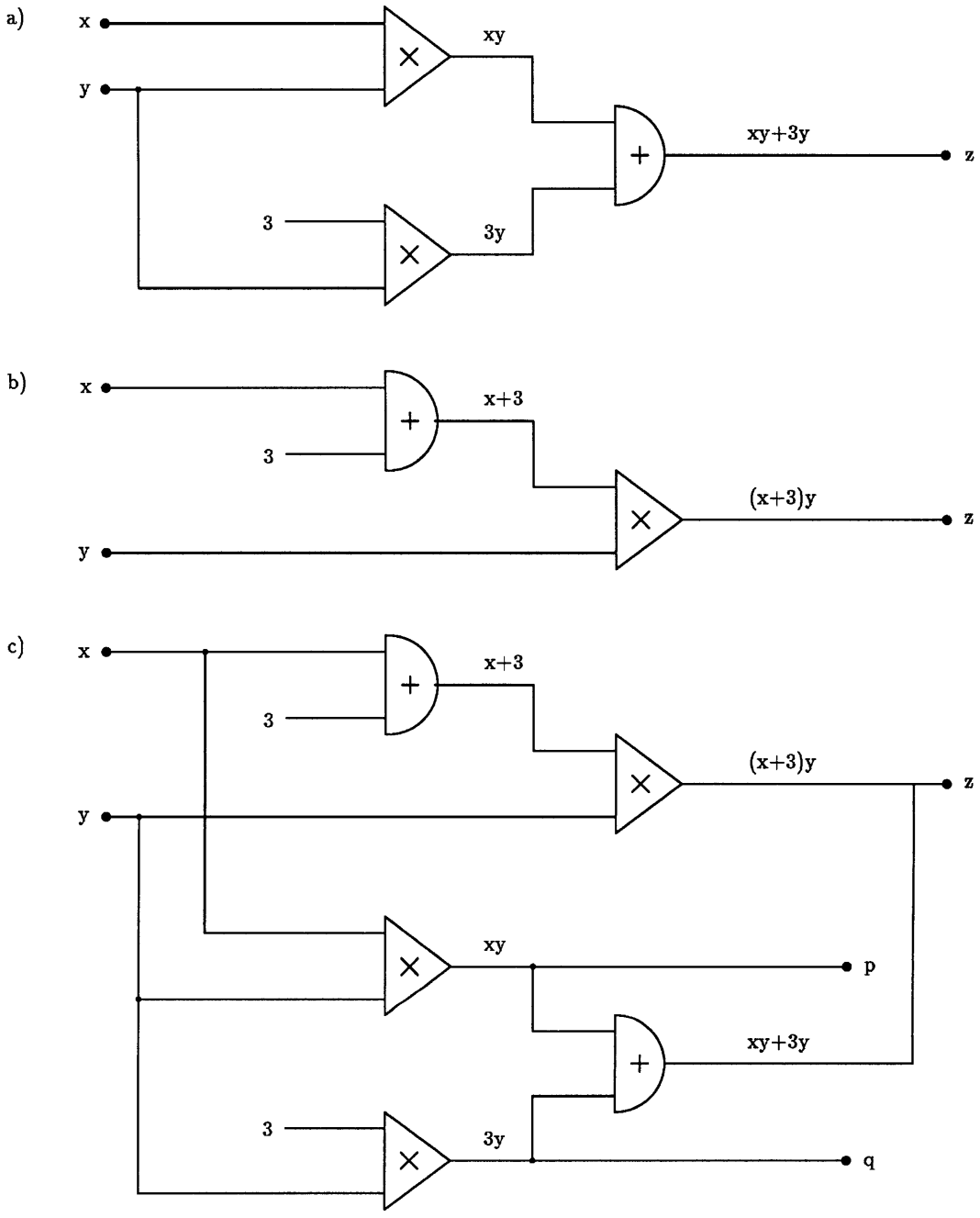

Figure 5.5: Distributivity

eliminated from the network. Given the values of any two of the three variables, the value of the third one can be deduced by local inference and propagation. If the value of the partial products is also needed or if it may be imposed by an other part of the network, a redundant network corresponding to the following system of equations may be designed to combine the various aspect of the problem and allow all potential computation to occur with no commitment as to which values will be available. This more complex network is presented in figure 5.5.c by superposing the previous two versions of the network. It corresponds to the following redundant system of equations:

$$\begin{cases} (x+3)y = z \\ p = xy \\ q = 3y \\ p+q = z \end{cases}$$

One can see from this example that a purely declarative approach is limited by the weakness of the resolution mechanism. In the absence of automated algebraic manipulation, the choice of the equations describing a given problem should be influenced by the structure of the underlying network. The resolution process may deadlock if no local inference can be performed after all the known values have been propagated. This problem can practically be avoided by providing redundant descriptions which reflect different formulations of the same system of equations.

5.2.6 Choice of the Equations

The method of resolution does not have the mathematical ambition of solving an arbitrary system of equations. It is often restricted by the locality of the inference process. The scope of inference may be extended by redundant views of a system of equations which introduce results of algebraic transforms in the network structure. In fact we shall see that the choice of the representational framework and the nature of the constraints involved in the description of flight paths minimizes the need for such redundant descriptions and allow a simple and efficient use of the resolution mechanism. Without necessarily affecting the termination of the resolution process, the choice of the equations influences the construction mechanism of the flight paths. The efficiency of the resolution method depends on the topology of the network which describes the system of equations.

Consider the simple case of an equality relation. Because of the transitivity of the equality relation, the number of wires which are necessary to express the equality of n variables need not exceed $n-1$ if a linear structure is chosen. In the worse case $n-1$ propagation steps are necessary to propagate the value to all the variables. If a central node is introduced n connections are needed, one from each node to the central node. Propagation requires two steps. If the transitive closure of the equivalence relationship is explicitly recorded, a fully connected set

of nodes has $\frac{n(n-1)}{2}$ connections and propagation requires one step. For a large number of nodes a star-shaped network with an auxiliary central node certainly provides the best compromise between complexity of the description and efficiency of propagation and resolution. This choice does not affect the termination of the resolution process.

5.2.7 Modularity

The resolution process is particularly well-suited to the case when the topology of the network reflects a high degree of modularity. Modularity reduces the synergy of the system and imposes a conceptual structure. A module is a portion of the system which performs a logical task or whose parts appears to be closely connected. By focusing on one module at a time it becomes tractable to reason about the details of its behavior, assuming that the peripheral parts with which the module interacts perform the expected tasks.

Redundant views of the module behavior are possible, though it would be too complex at the scale of the complete network. The designer has to organize the network at the scale of each module so as to exploit some algebraic properties of the system of equations under consideration in order to ensure the termination of the resolution process. The module boundaries are chosen so that modules are sparsely connected. To avoid deadlocks the designer has to check the behavior of inter-module loops assuming the correct functioning of each module in isolation. This method is easily applicable to a network which is organized as a chain of modules. It is the case for flight path plans which are built as sequences of states and operators.

5.2.8 A Layered Structure

The constraints are organized in superposed layers in order to take advantage of a modular description. In each layer, the constraints are defined in terms of the constraints of the underlying layers. Increasingly complex constraints can be constructed this way with an apparent simplicity. The degree of abstraction of the description increases as new layers of constraints are built on top of the existing ones.

Primitive constraints express and enforce simple algebraic relationships. They are the basic elements from which more sophisticated constraints are built. Compounded constraints combine primitive constraints and simpler compounded constraints. All these constraints provide the mathematical tools with which the geometric and kinematic properties of the flight paths can be expressed. Operators are defined in terms of these mathematical constraints. More complex maneuvers may be expressed as combinations of other operators.

As described in chapter 3, flight paths may be represented at various level of

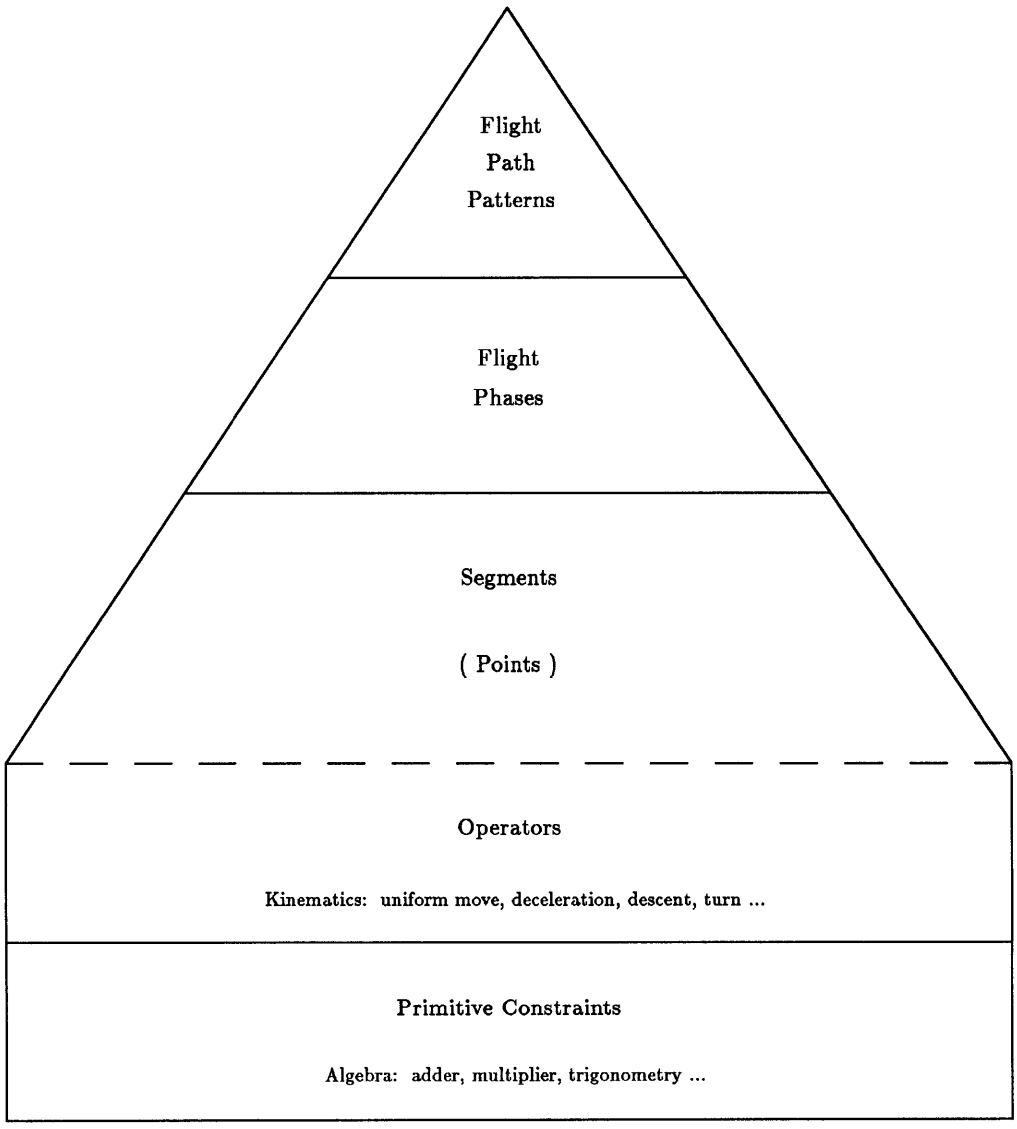

Figure 5.6: Structured Organizations

abstraction which are organized in a hierarchy. Segments are the basic descriptive elements. Each segment corresponds to the application of an operator between two states. This hierarchical description of flight paths provides a framework of knowledge representation for structured reasoning while an underlying construction mechanism achieves all the numerical computation automatically. For the purpose of software robustness and clarity of the programming style, the construction mechanism is structured and organized in layers. Figure 5.6 schematizes the structures of these organizations.

5.2.9 Consistency

The constraint paradigm and the propagation mechanism rely on intuitive declarative and operational principles. The behavior of elementary constraint devices, the structure of the network, the computational effects of the propagation mechanism, the limitations of the inference process can be easily visualized. By an analogy with electrical circuits, constraints may be viewed as ideal electrical devices which are connected by wires channelizing the flows of information. All devices potentially operate independently and in parallel.

The development of the constraint paradigm in the field of artificial intelligence was motivated by the need to analyze and design increasingly complex electrical circuits [69] [66]. Choosing the design parameters by solving the system of equations which results from the symbolic analysis of the circuit is in general untractable. This model of computation was initially developed to help electrical circuit designers choose the values of some parameters while the necessary numerical computation was performed in the background. Constraints may be used as a general framework for computer-aided design.

In this context, a constraint programming system should interact with the user and should maintain the consistency of the set of constraints [67]. A Truth Maintenance System (TMS) may be used to spot and track down inconsistencies [55]. Typically, dependency relationships are recorded and if an inconsistency arises, the conflicting facts and all conclusions which depend upon them are retracted. A comparable (but simpler) approach was taken for the design of our first prototype of a Flight Path Generator in the form of an Expert System using the logic programming language Prolog. Whenever a variable was instantiated as a result of local inference or propagation, consistency was checked. An attempt to instantiate a variable with two different values caused backtracking. The effects of the computation were undone back to the point where an alternative choice was possible. The instantiation strategy of the plan matched Prolog chronological backtracking. As indicated in the introduction, this software tool was used to improve and test the definition of flight path patterns.

A state-of-the-art constraint programming system has been developed in Lisp for a sequential implementation [68]. It provides the user with the tools to interact

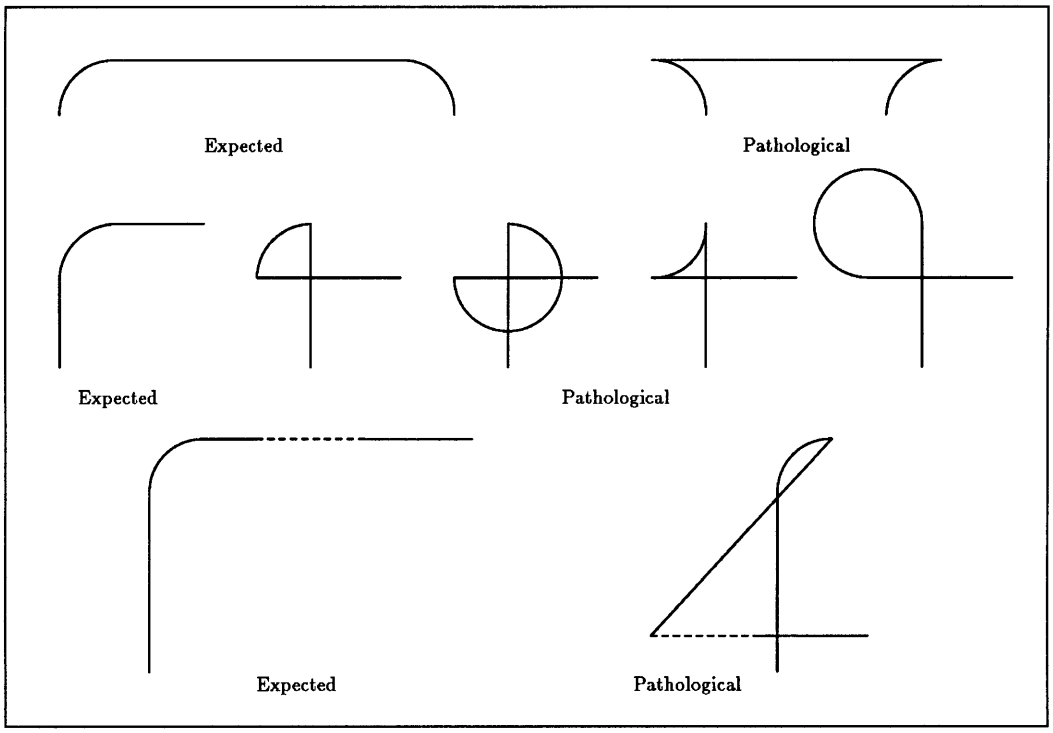

Figure 5.7: Examples of Pathological Cases

with the system so as to incrementally build and modify a network of constraints. All this machinery however is not used for the purpose of flight path generation. The definitions of the paths are initially constrained by a well-defined semantics and predefined patterns are used as flexible forms to build operationally feasible flight path plans. The availability of such a sophisticated programming system could have simplified the design of the patterns, but is useless at the stage of generation when speed is one of the main concerns.

No dynamical truth maintenance is performed and it is not checked that each individual constraint is satisfied at the level of the primitive operators. It is implicitly assumed that the system of equations is feasible and that no inconsistency may result from the resolution process. Special attention must be given to this issue when a network of constraints is designed. As it will be seen in 5.3.6, the specific nature of numerical constraints also involves additional problems to avoid introducing inconsistencies in the network.

However all the solutions which can be built from a given pattern do not necessarily correspond to a valid flight path. Figure 5.7 presents some typical examples of pathological cases where propagating the basic constraints imposed on the definition of the patterns leads to aberrations. The length of a leg may

be too short to accommodate the planned maneuvers, or the relative positions of some segments may not allow the insertion of an intermediate one so as to ensure the smoothness of the trajectory. Inconsistencies in the structure of the flight path plans are detected at a higher level of abstraction, mainly in terms of geometric properties. Paths have to be validated by a series of tests which occur during the construction process to filter out the ones which involve inconsistencies. In fact, it is often enough to ensure that the length of the time intervals during which operators are applied, and the geometric length of the segments in the case of straight-line motions, are positive quantities. Such basic conditions of path validity can only be checked once the paths have been at least partially built. If an aberration in the structure of a path is detected, new paths corresponding to different values for the degrees of freedom may have to be generated unless a satisfactory number of valid paths have already be generated in parallel with the one which is inconsistent.

A parallel implementation of the constraint propagation model using dataflow architecture computers is presented below. It provides a framework to define and run general-purpose networks of constraints and is used to quickly generate multiple flight path plans.

5.3 Dataflow Implementation

After an overview of the dataflow model of computation aimed at highlighting the similarities with the constraint propagation model, an implementation of the propagation mechanism in the programming language *Id* is presented. The implementation has a strong logic programming flavor which is a consequence of a previous prototyping of a Flight Path Generator in Prolog. The logic programming approach was appropriate for a declarative definition of the constraints. On the other hand the efficiency of the propagation mechanism depended on the choice of a propagation strategy and was restricted by the sequentiality of computation. Nested loops of propagation were used to give the priority to local propagation and inference before incrementally extending the range of propagation. The dataflow model of computation supporting a declarative language appear to be a natural way of expressing the notion of constraint and to implement the operational aspects of the propagation mechanism. The following references trace the history of dataflow computing: [54] [46] [48] [76] [57] [59] [51] [52].

5.3.1 Dataflow Graphs

A dataflow graph consists of operators connected by directed arcs that represent data-dependencies between the operators [54] [49]. An operator corresponds to a machine instruction such as addition, multiplication, array structure selection. Each operator may have one or more input and output arcs. Arcs may be

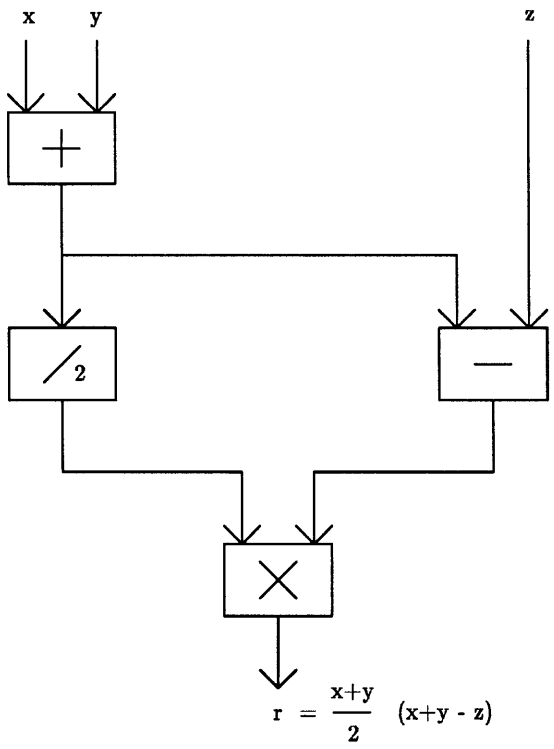

Figure 5.8: Dataflow Graph for an Arithmetic Expression

named in correspondence with the names of program variables. Any arithmetic or logical expression can be translated into an acyclic dataflow graph in a straightforward manner. An example of dataflow graph for a simple arithmetic expression is presented in figure 5.8. A fundamental difference with constraint networks is that arcs are directed whereas wires allow a bidirectional flow of information.

Data values are carried between operators on "tokens" which are said to flow along the arcs. This physical image is represented in dataflow machines by including a destination in the token that is the address of the instruction at the end of the arc. Execution is data-driven. Basically, an operator is ready to fire and to perform the designated operation when tokens are available on all its input arcs. This means that all its operands have a known value. Firing an operator involves consuming all its input tokens, executing an instruction with the values carried on the tokens, and producing a result token on each output arc.

Parallelism and determinacy are two key properties of dataflow graphs. Operators may potentially fire in parallel unless there is an explicit data dependence between them. The result does not depend on the relative order in which potentially concurrent operators fire. As in the case of constraint networks, operation can potentially occur in parallel and asynchronously as soon as a sufficient number of values are present at the pins. Another fundamental difference is that each operator corresponds to a definite machine instruction which imposes the direction of the flow of computation and does not correspond to a declarative statement.

Because of parallel invocations and recursion, a function can have many simultaneous activations. One needs a way of distinguishing tokens within the graph representing a function that logically correspond to different activations. Instead of copying the entire graph of the function body for each activation, tokens are tagged with a context identifier that specifies the activation to which it belongs [51]. The dataflow graph for a function corresponds to the fixed code of its definition. A token carries a tag and a datum. The tag is a continuation which indicates what is to be done with the datum. It consists of a dynamic context that specify the frame for a particular invocation of the function, the address of the destination instruction, and the port identifying the inputs of the instruction. The datum is a value or the descriptor of a structure (a pointer to the structure). The activation rule of the operators has to be adapted to account for this more detailed description of the tokens. An operator is ready to fire when a set of input tokens with matched tags have arrived at its input arcs.

Dataflow graphs constitute a machine language in the sense that they are directly executable by dataflow architecture computers. Dataflow graphs are consequently the target for compilation. The computing facilities we use in our project of developing a prototype of Flight Path Generator include an emulation of the *MIT Tagged-Token Dataflow Architecture*. This tool simulates a machine for executing dataflow graphs with hardware support for data-driven instruction scheduling (unlike the sequential Program Counter-based scheduling of traditional von

Neumann machines).

5.3.2 A Programming Language: *Id*

Id is a high-level, non-strict language with fine-grained parallelism and determinacy implicit in its operational semantics. The core language is functional. Currying of higher-order functions with partial application is supported. [63]

Id insulates the programmer from the underlying machine architecture in terms of number and speed of processors, topology, and speed of the communication network. Parallelism is implicit. The programmer does not have to partition the program into parallel tasks or to annotate it to indicate opportunities for parallel execution. *Id* programs are determinate. Determinacy of functional languages is guaranteed by the Church-Rosser property. The programmer does not have to explicitly manage scheduling and synchronization. The computed result depends only on the program input and is independent from machine configuration and load.

I-structures are a way of introducing a limited notion of state into dataflow graphs without compromising parallelism or determinacy [47] [50]. Semantically an I-structure may be viewed as a dynamically allocated array of logical terms. The single-assignment rule avoids inconsistent instantiations and an attempt to overwrite a non-empty location causes a run-time error. The deferred read avoids read-write races. A consumer who tries to read an I-structure location is made to wait until the location has been filled in. Logical terms can not be tested for instantiation using for instance the instructions *var* and *nonvar* as in Prolog. The restrictions "write-once", "deferred read", and "no test for emptiness", ensure that the language remains determinate. Besides the non-strictness of the I-structures allows the consumer of an I-structure to start working on it before the producer has finished filling in all the locations. As a consequence of the introduction of non-functional feature such as I-structures, the language loses referential transparency with all the attendant implications on the ability to perform program transformations. For this reason functional data-structure abstractions should be defined to encapsulate the usage of I-structures.

Id programs can easily be compiled into dynamic dataflow graphs. Its essentially functional and determinate semantics enables compiling optimizations such as loop-constant detection, constant folding, inline substitution and fast function calls. Besides compiling is simplified since the object code is independent from the machine characteristics. [71] [72]

5.3.3 Reconciling Dataflow Graphs and Constraint Networks

Dataflow and constraint propagation are two models of computation which

	Dataflow Graphs	Constraint Networks
Connections	directed arc	bidirectional wire
Nodes, operators	mathematical function	mathematical relation
Pins	distinction between input and output of an operator	similar pins
Flow of computation	fixed	unspecified

Figure 5.9: Conceptual Differences

may be associated with the same imagery as a consequence of the underlying network structure which characterizes both. Typically, a structure built from interconnected operators supports asynchronous concurrent activities. Information is propagated internally in the form of values, the availability of which triggers the firing of operators.

The use of a common terminology reinforces the intuition of similitude though the two approaches differ on several fundamental aspects. An ordinary misconception about dataflow architecture is that the graphs represent an interconnection of hardware modules. The conceptual differences between the two models are summarized in the table of figure 5.9. Constraint networks are more general than dataflow graphs in the sense that they offer more freedom at the operational level as much as at the representational level. A dataflow graph can be obtained from a constraint network by restricting the network to a predefined behavior. The opposite transformation happens to be needed. The goal at this point is to express constraint networks and to emulate their functioning by the means of dataflow graphs so that they can be run extremely fast by dataflow architecture computers.

The proposed solution is the following. A constraint network is built as a superposition of dataflow graphs which are combined by the intermediary of a synchronization mechanism. A dataflow graph is provided for each possible flow of computation. The synchronization mechanism detects the availability of data, dynamically selects a dataflow graph, and ensures the uniqueness of the graph being operational.

Consider the case of an operator of arity three which needs the value of two

out of its three arguments to be fired and be able to infer new information. For example an adder has two operands and a sum; the value of the third one can be inferred from the values of any two of the arguments. Three complementary dataflow graphs reflecting three modes of functioning are necessary to describe its behavior in all possible situations. This is the case for all reversible binary operators. In the case of an operator with m pins which can operate as soon as any n values are available, the number of necessary graphs is as high as the number of combinations C_m^n. Let p be the number of these operators which are connected together and organized in a network, and assume for simplicity that all operators are of the same type. Out of the $(C_m^n)^p$ a-priori possible behaviors, some are illegal in a dataflow graph structure because they involve connecting two outputs or because several inputs are isolated and can not receive any token or value. Defining a dataflow graph for each possible behavior of a huge network of constraints and synchronizing the behavior of all the parts would involve a combinatorial explosion and become untractable.

Modularity again plays an important role at this point. The user should not be expected to define a network at the basic level, in terms of operators, arcs, partial graphs, and to synchronize their behavior. Instead an elementary language of constraints is provided to insulate the programmer from the structure of the graphs and allow a high-level description of the network. Synchronization occurs with the finest possible granularity; it is distributed and performed automatically. A set of primitive operators which incorporate their own synchronization mechanism is provided. As explained earlier, primitive operators allow expressing simple algebraic relationships based on arithmetics and trigonometry. A network of constraints is built for each pattern from these basic elements.

5.3.4 Proposed Dataflow Implementation

Figure 5.10 shows how an adder may be synthesized from the dataflow graphs associated with one addition and two subtraction instructions. The bus at the left of the schema may be viewed as set of potential lines, where the value of the potential reflects the values of the variables at the pins. The synchronization unit detects the presence of non-default potential values or the availability of tokens on the bus. After performing some logic on these data the synchronization unit may trigger the activation of at most one of the three instructions. This may happen only if the corresponding operator is ready to fire with matched tokens on all its inputs. Dataflow operators would need a special trigger input. The instruction is executed and the result output token is delivered on a line of the bus or equivalently the potential of the line is set the the new result value. Inconsistencies may be detected by comparing the result value to the bus value before equating them.

The synchronization unit is really the part which determines the actual be-

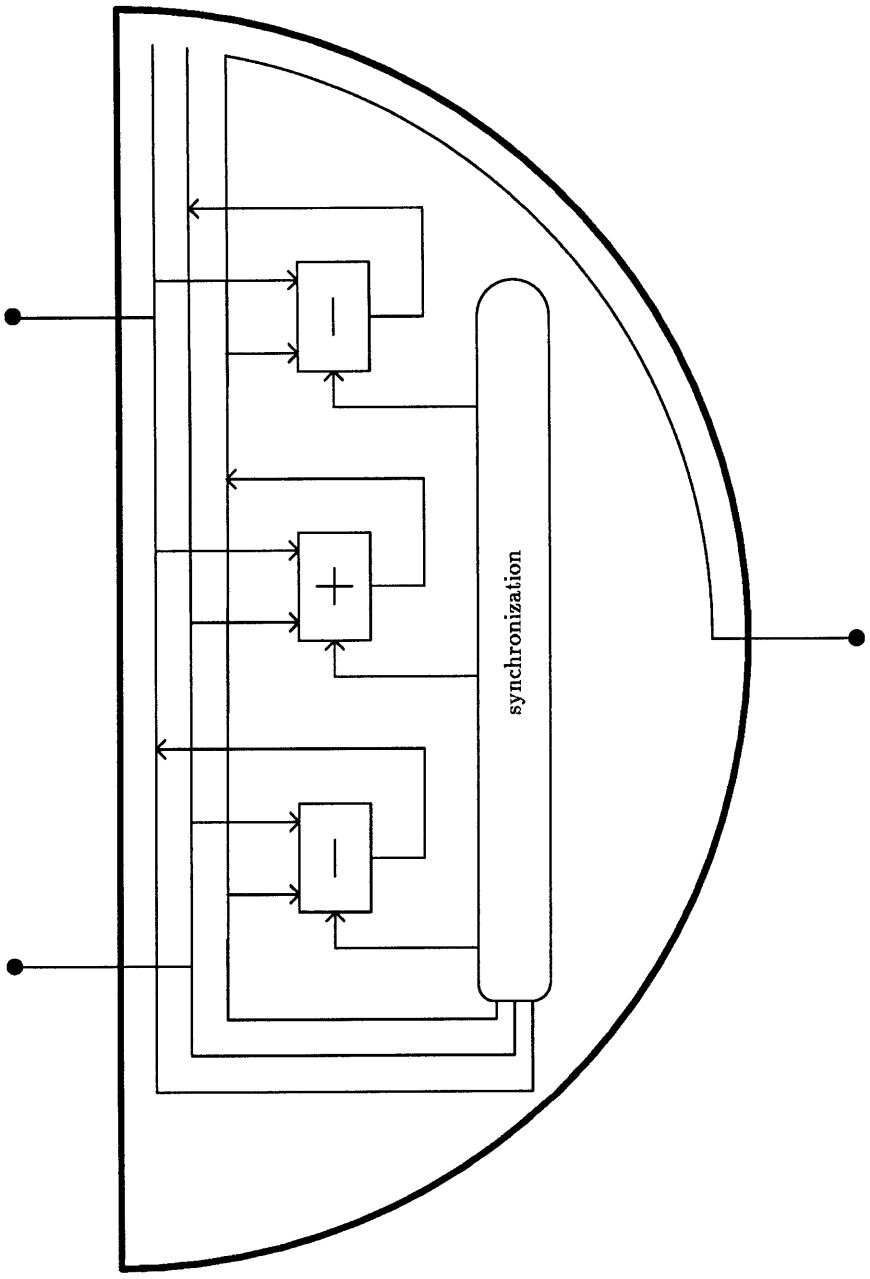

Figure 5.10: Synthesis of an Adder

havior of the device (which instruction is activated and when). Whether a change in a value at one of the pins will trigger another activation of an instruction is a question which involves the global stability of the network. The implementation choice is the following. The variables at the pins are assumed to be initially uninstantiated. As soon as two of them (in the case of an adder) are instantiated, the synchronization unit triggers the activation of the appropriate instruction so that the two values are used to infer the third one. Activation occurs exactly once. The fact that the third variable might be instantiated by propagation from the rest of the network while its value is being computed locally by the device is a potential source of inconsistency in the network. As noted in 5.2.9, dynamical truth maintenance is not used for the purpose of fast path generation. Most of the features developed to manipulate constraints and operate on networks of constraints [68] may be adaptable to this dataflow implementation. However the definition of the operators and the notion of consistency should be extended to deal with the problems imputable to the manipulation of real numbers and not only integers.

5.3.5 *Id* Implementation

The commitment to a single activation of each device and the assumption of consistency allow a simple implementation of the constraint propagation model in *Id*. After praising the virtues of *Id* for its clean determinate semantics, we are going to introduce shared data structures as a means of expressing and controlling non-determinacy. Non-determinacy is inherent to the functioning of the operators since the way they operate depends on the availability of information which arrive asynchronously. In traditional programming languages, the synchronization of the operators could have been achieved by using loops which regularly test for the availability of a sufficient subset of values at the pins. At a lower level of implementation, the arrival of new values could cause an interrupt which triggers the activation of the operator.

To take advantage of the data-driven instruction scheduling of the dataflow architecture, switches are used which passively detect the availability of values at the pins and direct the flow of computation depending on the asynchronous arrival of data. Figure 5.11 presents the code of some synchronization primitives. Their role is to set a flag to a value which depends on the subset of variables which have been instantiated. Setting the flag triggers the activation of the operator. The value of the flag is used in a "case" expression to switch to the right dataflow graph and execute the appropriate instruction. Figure 5.12 presents the code for the adder which uses the synchronization flag. Less conventional operators may need a specific synchronization procedure. Using the same two-step approach as for the adder, arbitrarily complex operators may be design and synchronized.

"Side-effects" are only used in a very restricted way within the scope of a

```
% Gate
% "gate" returns the value of "expression" after "synchro" has received
% a value. In the case of an eager evaluation, arguments (for which the
% function is strict) are evaluated in parallel with the function call.
% Thus "expression" is evaluated independently of the synchronization.
% "gate" does not synchronize the side-effects which may be involved by
% the evaluation of "expression".

def gate synchro expression =
  if synchro == synchro
     then  expression
  %second arm of the conditional to avoid returning "void"
  else  expression;

def wait_for x =
  gate x true;

% Non-Determinate Conditional

def ndc_1_out_of_2 x1 x2 =
  {flag = allocate_cell 'd;
   if (wait_for x1)
      then  write_cell flag 1;
   if (wait_for x2)
      then  write_cell flag 2;
   in
     read_cell flag};

def ndc_2_out_of_3 x1 x2 x3 =
  {flag = allocate_cell 'd;
   if (wait_for x1) and (wait_for x2)
      then  write_cell flag 1;
   if (wait_for x2) and (wait_for x3)
      then  write_cell flag 2;
   if (wait_for x3) and (wait_for x1)
      then  write_cell flag 3;
   in
     read_cell flag};

% Synchronization of the elementary constraint modules

% like absolute-value
def synchronize_non_reversible_unary_operation op1 r =
  wait_for (value op1);

% like minus
def synchronize_reversible_unary_operation op1 r =
  ndc_1_out_of_2 (value op1) (value r);

% like sum
def synchronize_reversible_binary_operation op1 op2 r =
  ndc_2_out_of_3 (value op1) (value op2) (value r);
```

Figure 5.11: Synchronization Primitives

```
def sum op1 op2 sum =
  {synchro_flag = synchronize_sum op1 op2 sum;
   in
     {case synchro_flag of
         1 = sum_1 op1 op2 sum
       | 2 = sum_2 op1 op2 sum
       | 3 = sum_3 op1 op2 sum}};

def sum_1 op1 op2 sum =
  {v_op1 = value op1;
   v_op2 = value op2;
   v_sum = v_op1 + v_op2;
   in
     instantiate sum v_sum};

def sum_2 op1 op2 sum =
  {v_op1 = v_sum - v_op2;
   v_op2 = value op2;
   v_sum = value sum;
   in
     instantiate op1 v_op1};

def sum_3 op1 op2 sum =
  {v_op1 = value op1;
   v_op2 = v_sum - v_op1;
   v_sum = value sum;
   in
     instantiate op2 v_op2};
```

Figure 5.12: Code for an Adder

synchronization procedure. The procedure allocates a cell to be used as a flag and has the privilege of multiple write operations. In general, the first value to be written in the cell is read and returned functionally as the result of the procedure evaluation. Other values may be subsequently written in the cell but they are ignored. If for hardware scheduling reasons the read operation happens to be delayed, the second written value may be read. The result is actually the same as if the process delivering the first value had been slower than the one delivering the second one and that what happened to be the second value was the first to be written and read. This situation may occur only if values are available on two subsets of pins and the operator may be fired in two different ways, both leading to consistent results.

All the state variables and the operator parameters are implemented as locations of shared data structures which may potentially be read and written by any of the operators which have access to them. In general, each location is written only once, but multiple consistent write operations are acceptable as long as they are consistent in the sense defined below. Unlike the situation in hardware, there is no need to replicate the devices. A constraint device is treated as a procedure call and the variables between which it imposes a constraint are passed as arguments. By tagging the tokens, replication of the graph is avoided.

```
def synchronize_prod op1 op2 prod =
  {flag = allocate_cell 'd;
   v_op1 = value op1;
   v_op2 = value op2;
   v_prod = value prod;
   w_op1 = wait_for v_op1;
   w_op2 = wait_for v_op2;
   w_prod = wait_for v_prod;
   if w_op1 and w_op2
      then
     write_cell flag 1;
   if v_op2<>0 and w_prod
      then
     write_cell flag 2;
   if v_op2 == 0
     then
     write_cell flag 4;
   if v_op1<>0 and w_prod
      then
     write_cell flag 3;
   if v_op1 == 0
     then
     write_cell flag 4;
   in
      read_cell flag};
```

Figure 5.13: Code for the Synchronization of a Multiplier

5.3.6 Representation Issues

An inconsistency in the network of constraints arises if two devices try to impose different values on pins which are directly connected by a wire, or equivalently if is there is an attempt to match logic variables which have different values or to instantiate a logic variable more than once.

The problem of inconsistency detection is more general than it may look at first sight. The following sentences refer to the terminology of mathematical logic. Patterns are literal expressions involving constants and variables. Given a set of patterns, unification is the process of finding a substitution that make the patterns equal [62] [64]. A substitution step consists of binding a variable to a term and substituting for the term all instances of the variable in the patterns. Practically, unification involves binding variables to constants and equating variables or functions of variables. Patterns match if they can be unified. They are equal if they are identical strings of characters. Unifying expressions that involve numerical constants raises theoretical representation issues [56]. Two constants should match if they may be considered as being equal from the point of view of a semantic identity independently from the syntax, i.e. if they actually correspond to the same number but do not necessarily use the same representation.

Matching patterns requires being able to recognize that objects are the same even if they do not look alike. This is often a problem even with a consistent mode of representation. For instance, reduction to the same denominator or to the same exponent may be a prerequisite for comparing numbers. Similarly 0.999...[9]... and

1 represent the same decimal number. Terms have to be converted into a canonical form before being compared. Comparing computer representation of numbers may involve type coercion. In the problem at hand, we are only concerned with the decimal representation of real numbers as used in physics.

The real issue is precision. Evaluating mathematical expressions which are logically equal may lead to different results because of the limited precision of computation. Errors are accumulated and amplified by a cascade of operations. Numerical analysis is not an issue here since devices operate asynchronously and have enough time to reach a desirable precision. The problem is to recognize that separately calculated results are intended to be the same though they differ. Since variables are involved in several equations, their values may be computed by independent means and not exactly satisfy an equality because of a limited precision. Whether this should be considered as an inconsistency or not depends on the amplitude of the discrepancy. The choice of the precision is application-dependent. It should be taken into account the precision of resolution of the equations by each operator, the interdependence of the equations in order to set an upper bound on the cumulated error, but also the expected accuracy of the result and the level of noise of the data.

Apart from the issue of consistency, another problem arises when the behavior of an operator depends on specific values at its pins. The continuity of the functions involved avoids the problems associated with thresholds or isolated points. Consider the case of a simple multiplication operation. Zero plays a special role since it is the absorbent element. "$0 \times 3 = y$" implies "$y = 0$", obviously. "$0 \times x = y$" implies "$y = 0$"; the relation is satisfied independently of the value of x which remains unspecified: "$x = \perp$" ($\perp$ is the "bottom" symbol indicating an absence of information). "$xy = 0$" implies "$x = 0$ *or* $y = 0$". This level of reasoning which requires the notion of hypothesis is not considered here.

The existence of these particular cases explains that the synchronization of a multiplier as it appears in figure 5.13, is a bit more sophisticated than for an adder. The problem of precision is illustrated by the following example:

$$0 \times x = 0 \;\Rightarrow\; x = \perp$$

but

$$\epsilon_1 \times x = \epsilon_2 \;\Rightarrow\; x = \frac{\epsilon_2}{\epsilon_1}$$

In the second case, instead of leaving x undetermined, an arbitrary and meaningless value is imposed which may conflict with a subsequent instantiation of the variable x . This problem arose during the development of the Flight Path Generator. A downwind leg is parallel to the runway centerline. According to our choice of spatial coordinates, the y variations along this leg are equal to zero.

```
def prod op1 op2 prod =
  {synchro_flag = synchronize_prod op1 op2 prod;
   in
     {case synchro_flag of
         1 = prod_1 op1 op2 prod
       | 2 = prod_2 op1 op2 prod
       | 3 = prod_3 op1 op2 prod
       | 4 = prod_4 op1 op2 prod}};

def prod_1 op1 op2 prod =
  {v_op1 = value op1;
   v_op2 = value op2;
   v_prod = v_op1 * v_op2;
   in
     instantiate prod v_prod};

def prod_2 op1 op2 prod =
  {v_op1 = v_prod / v_op2;
   v_op2 = value op2;
   v_prod = value prod;
   in
     if not (equivalent_to_precision_error v_op2)
        then
       instantiate op1 v_op1};

def prod_3 op1 op2 prod =
  {v_op1 = value op1;
   v_op2 = v_prod / v_op1;
   v_prod = value prod;
   in
     if not (equivalent_to_precision_error v_op1)
        then
       instantiate op2 v_op2};

% zero is the absorbant element
def prod_4 op1 op2 prod =
  {v_prod = 0;
   in
     instantiate prod v_prod};

def equivalent_to_precision_error x =
  (abs x) < (precision_error_threshold 'd);

def precision_error_threshold d =
  1e-6;
```

Figure 5.14: Code for a Multiplier

The relationship "$\delta y = \delta l \sin\theta$" holds, where δl denotes the length of a segment corresponding to a straight-line motion. As a consequence of perfectly acceptable precision inaccuracies, neither δy nor $\sin\theta$ were exactly equal to zero, and instead of leaving δl unspecified so that it may be instantiated as a result of the equations "$\delta x = \delta l \cos\theta$" and "$\delta x = v\delta t$" , it was instantiated to an arbitrary value which even happened to be negative. This example should justify the implementation of the multiplier as it is presented in figure 5.14.

5.3.7 Parallelism Profile

The parallelism profile of an execution gives the number of operations which can be executed at each time step. *Id World* is an integrated programming environment for *Id* and the *Tagged-Token Dataflow Architecture*, which provides the facilities to measure and plot parallelism profiles, instruction counts, memory requirements, and other statistics for the emulated machine. An asynchronous dataflow execution is modeled by the means of synchronous execution units under the following assumptions. Every operation takes one time unit. Operations are fired eagerly. Memory and communication resources are not limited. Communication delays, memory and pipeline latency, and the number of processors can be adjusted to simulate various hardware design choices and machine configurations. All the following examples assume the characteristics of an ideal machine except for the number of processors which is not considered infinite. The influence of the number of processors is analyzed. The program which is run with various machine configuration is composed of six hundred procedures for a size of five thousand lines of *Id* code.

Parallelism Profiles for the Generation of a Flight Path Plan

Figure 5.15 gathers the parallelism profiles for some executions corresponding to the generation of a flight path plan. The number of arithmetic and logic (ALU) operations is recorded as a function of time. (When two curves are plotted on the same graph, they are the envelopes of the "high frequency" fluctuations of the number of simultaneous ALU operations.) The flight path which is generated is an instance of the pattern "arrival-trombone" described in chapter 4. Whereas the generation process depends on the choice of a pattern, it is quasi-independent from the values of the degrees of freedom.

A lot of parallelism is available during the construction of the constraint network. Memory is allocated for data structures which include state variables and operator parameters. The linkage of these cells to build the network is embedded in the dataflow graph. The first narrow peak of the parallelism profiles corresponds to a brief initialization phase. The constraints which are inherent to the structure of the pattern can be applied immediately. For example, the flight direction along the downwind leg is known as soon as the runway is fixed. The

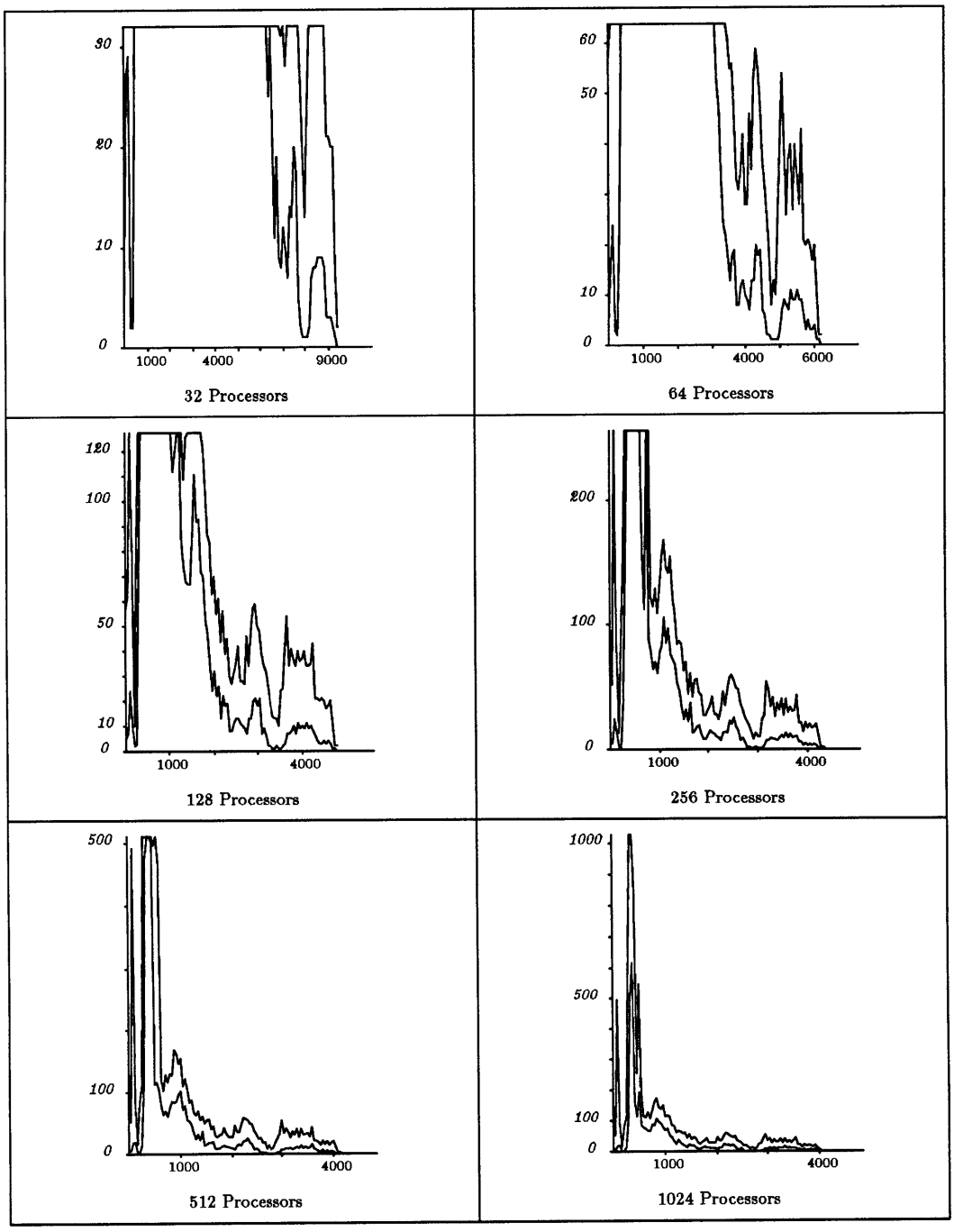

Figure 5.15: ALU Operations Profiles for the Generation of a Flight Path Plan

angle of the turn from the downwind leg to the base leg is instantiated to this value plus or minus $\frac{\pi}{2}$ depending on the side of the runway on which the approach is performed. Similarly, the values of the maneuver characteristics corresponding to a given aircraft model or type are fetched from a database implemented in an object-oriented style and imposed as constraints on the paths. The amount of information which is initially provided explains the concentration of computation at the beginning of the construction process.

A maximum of one thousand and fifty processors may potentially be kept busy simultaneously when all the parallelism which is inherent to the program is exploited. When less processors are available the machine saturates and some parts of the execution have to be delayed. A total of two hundred and fifty thousand operations corresponding to the area under the curves is executed.

Influence of the Number of Processors

Figure 5.16 shows how the number of processors influences the execution time. The "length of the critical path" which appears on the y-axis, characterizes the duration of the execution. It can be defined as the time, measured from the beginning of the execution, after which the parallelism profile indicates that no more operation is executed. Remarks about this definition appear below. In the case of an idealized machine with an infinite number of processors, the critical path length is an intrinsic characteristic of the program.

The curve decreases asymptotically towards four thousand time units as the number of processors tends towards infinity. At most six percent may be gained by increasing the number of processors beyond 256.

Parallel Flight Path Generation

Figure 5.17 shows the parallelism profiles when four paths of the same nature are generated in parallel. With an infinite number of processors the parallelism profiles is the superposition of four similar profiles. If only 256 processors are available the execution time is increased by fifty percent when four path of the same nature instead of one are generated concurrently. The strategy of path generation depends on this result.

Since it is uncertain that the first generated path be conflict-free, it may be advantageous to generate several paths in parallel from the beginning. On the other hand attempting to generate a large number of paths would monopolize the computational resources and delay the choice of a path. A highly parallel computer such as the *Connection Machine* would favor the simultaneous generation of many paths [58] [70] [53]. A machine which offers less parallelism with potentially faster circuitry would favor the generation of clusters of flight paths (certainly less than ten paths with 256 processors). With a sequential implementation, the logic of

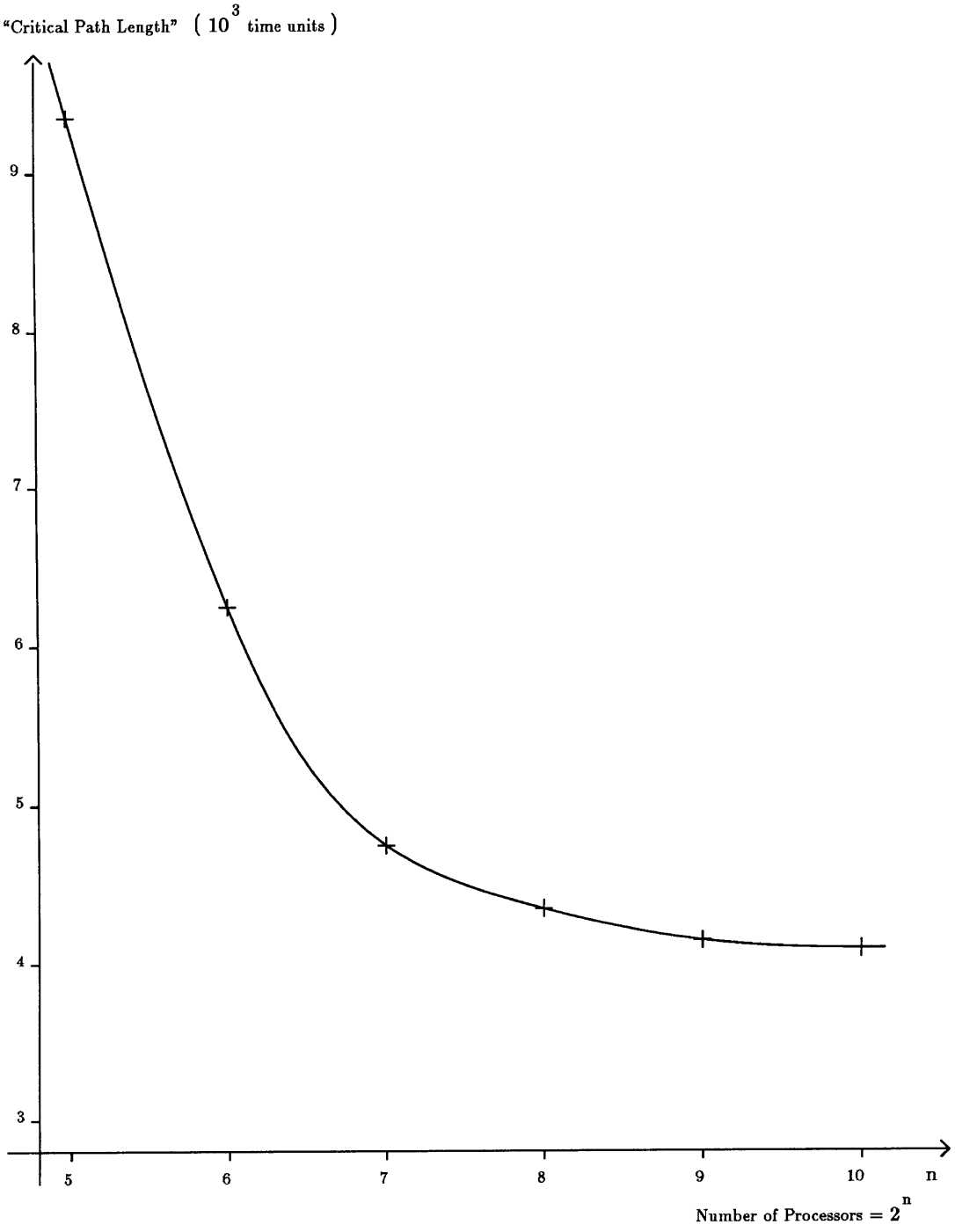

Figure 5.16: Critical Path Length as a Function of the Number of Processors

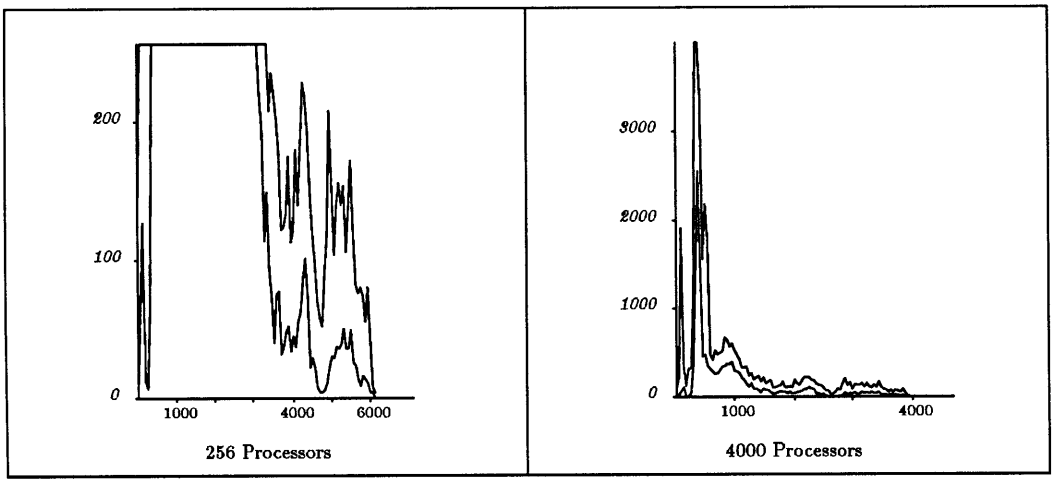

Figure 5.17: ALU Operations Profile for the Generation of Four Flight Path Plans in Parallel

path generation should focus on minimizing the number of paths which have to be generated before finding a satisfactory one.

Use of Real Hardware

The *Monsoon* dataflow architecture machine which is presently being constructed at the *MIT Laboratory for Computer Science*, deliverable in 1991, should have 256 processing elements and provide a 2 BIPS peak (1 BIPS sustained) computing power. The details of this architecture are described in [65]. An early prototype of the machine with a restricted number of processors is available at the time this report is written.

Generating one path with the completed machine would require approximately four micro seconds. On a computer which would be even four orders of magnitude slower, generating a feasible flight path plan would not require more than a few hundredths of second. If one second is allowable to generate and choose a conflict-free feasible path, such a speed gives sufficient leeway to generate a reasonable number of paths and detect potential conflicts.

Remarks about the Notions of Critical Path and Non-Determinism

The critical path length can be intuitively defined as the time of computing activity of an execution. Then, it can be easily measured as the duration of the period of non-zero parallelism profile.

The "critical path" may be defined as the longest chain of data dependencies

in the program. With an infinite number of processors, the length of this chain corresponds to the time between the beginning of an execution and the delivery of the result. Detecting that a result is available may involve technical difficulties when it is a structured object which is incrementally built, but let us ignore this fact. With a purely functional language, the portion of an execution which is still computing after a result has been delivered can be aborted. This is obviously not the case when side-effects are allowed, since the value of the result may be subsequently modified. With *Id* programs, the only issue is that an attempt of multiple writes on an I-Structure may cause a run-time error which invalidates the result. The time of result delivery appears as a notion which should be handle with care when the language is not functional. Though these two definitions are usually considered to be equivalent, they conflict when non-strictness or non-determinism are introduced.

With a non-strict language like *Id*, a result may be delivered as output before the computation has terminated. The rest of the computation simply cleans the well-behaved dataflow graph by removing the tokens which are still in transit [72]. The duration of hardware activity is dissociated from the time of result delivery since the longest chain of data dependencies may be shorter than the period of non-zero parallelism profile. Consider the situation where the numerical result of a first program is fed into a second one, and assume that the lengths of the critical paths are known for both. The result of the first program may be delivered to the second one before its execution terminates. Note that if the result was a non-strict data structure such as an I-structure, the second program could have started using portions of it even before it is entirely filled. If the definition relying on result delivery is used, the length of the critical path for the composition of the two programs is equal to the sum of the lengths of the two critical paths. If the definition relying on computing activity is used, this sum is only an upper bound.

Non-determinism questions the notion of dependency itself. *Id* programs are determinate since the language semantics guarantees that the results of different executions of the same program with the same inputs are identical, independently from the machine configuration or workload. These programs are non-deterministic in the sense that the order in which instructions are executed is not specified by the programmer, and may vary from one execution to another. The dataflow graphs embed the data dependencies between the instructions. Let us call this form of non-determinism "temporal non-determinism". In the case of the constraint propagation mechanism however, there is no predefined order of computation. Not only the time of execution, but also the nature of the instructions are unspecified. Consequently the static expression of dependency is not sufficient to characterize the computation which may result from purely declarative statements. This more general notion of non-determinism may be called "operational non-determinism".

An important question is to know whether operationally non-deterministic pro-

grams are determinate. The intuition is that they are not in general. However, the generation of flight paths by parallel constraint propagation can be viewed as determinate. The result of two executions starting with the same set of constraints are such that the state variables and the operator parameters are instantiated in both executions to values which do not differ by more than the precision error of the computation (as it appears for instance in the code of a multiplier in figure 5.14). This means that without being strictly equal, the results are at least consistent. For practical purposes, the two generated flight paths are identical.

The networks of constraints are designed for each pattern with special attention to avoid deadlocks and inconsistencies. This task is made easier by the layered organization of the constraints and by the modularity of their descriptions.

5.4 Conclusion

Flight paths are defined declaratively by an accumulation of constraints. The definitions and the software implementation are easily adaptable to a specific terminal area organization or to new aircraft performance and maneuver characteristics. The constraint propagation model of computation provides a framework to efficiently build flight paths by value propagation in a network of constraints. Constraints may be combined to define compounded constraints with an arbitrary complexity. The numerical computation is then performed automatically by the underlying mechanism which dynamically chooses the flow of computation depending on the availability of data. An electrical analogy makes the model intuitive and easy to manipulate so as to avoid deadlocks of the local inference process. A layered organization of the constraints enhances modularity and increases the flexibility and the expressivity of the constraint paradigm by introducing the opportunity for abstraction. At the bottom of the structure, primitive constraints provide the mathematical tools of arithmetics and trigonometry to express higher level geometric or kinematic properties. Consistency of the trajectories is maintained at the macroscopic level of path structure.

Similarities between dataflow graphs and constraint networks in terms of concurrent asynchronous activity and graph structure motivated the choice of dataflow architecture computers to generate flight paths by constraint propagation in order to match the programming paradigm and the architecture. In spite of a common imagery, dataflow graphs and constraint networks differ essentially about the determinacy of the flows of information. A constraint network may be implemented as the superposition of dataflow graphs supported by a synchronization mechanism. Synchronization is distributed and ensured by each primitive operator. The detailed example of the synthesis of an adder has been presented.

Non-determinacy is introduced in the clean functional semantics of the programming language *Id* by providing shared data structures. The representation and precision issues imputable to the manipulation of real numbers have required

special attention to avoid erroneous inference steps which would be the source of inconsistencies. Parallelism profiles show that a substantial amount of parallelism can be exploited during the generation of a flight path. The influence of the machine configuration in terms of number of processors was highlighted. For a machine with a couple hundreds of processing elements, a reasonable generation strategy is to build small clusters of operationally feasible flight paths in parallel. Before being displayed to an air traffic controller, these feasible flight paths have to be tested for conflict.

Chapter 6

Conflict Detection

The operationally feasible flight path plans which can be quickly generated by constraint propagation have to be tested for conflict so as to provide controllers with a variety of paths which are free of all violation of separation criteria. This chapter presents some tools for a general-purpose conflict detection which take advantage of the hierarchical representation of the paths. Symbolic reasoning and mathematical algorithms are combined into a flexible implementation. The efficiency of the detection may be enhanced by the use of some specific knowledge about the structure of the paths.

6.1 Methodology

The choice of the detection strategy is based on the observation of conflict sparsity expressed in the following fundamental assumption: in the process of detecting conflicts between two trajectories, most elements of which the trajectories are composed at a given level of abstraction are conflict-free. The strategy used for conflict detection consists of recognizing quickly and at low cost that big portions of trajectory are conflict-free. Only the elements at a given level of abstraction which could not be proved to be conflict-free are subsequently considered. According to the previous fundamental assumption, only a minority of these elements should have to be taken into account for further investigation. This strategy is applied recursively, focusing on the elements which are potential sources of conflict.

As explained in chapter 3, a flight path is represented with degrees of abstraction which are organized in a hierarchy and give an increasingly detailed description. Considering a flight path as a tree structure whose leaf-elements are segments, the branching coefficients are finite since every element is composed of a finite number of subelements. Segments however are analytic descriptions of elementary pieces of trajectory and consequently are implicitly composed of an infinite number of points. To ensure the termination of the recursive method,

analytic or numerical methods must be provided to deal with the mathematical representation in terms of segments and points.

Perfection is sought because of the primary concern for safety. Conflict detection requires a zero-default strategy and not an incremental improvement strategy which would consist of reducing the probability of conflict below a tolerance threshold. Trajectories are considered as conflict-free when it can be proven that they do not involve any conflict, not when no conflict can be detected for a given computational expense.

Conflict detection may be viewed as a succession of "cascaded" filtering processes using:

- high-level properties of the flight path patterns
- a geometric filtering at intermediate levels of abstraction
- analytic algorithms of kinematics to detect conflicts between segments
- a time-simulation of the trajectories

6.2 Notations and Definitions

6.2.1 Separation Criteria

A set of rules, known as Separation Standards, have been devised to ensure the safe separation of aircraft. The notion of Separation Standards and the separation criteria they impose are described in appendix A. An innovative model of aircraft interaction is also introduced. In the terminal area Separation Standards may be summarized by the following rules:

- minimum horizontal separation of three nautical miles
- minimum vertical separation of one thousand feet

Horizontal and vertical separation are decoupled and at least one of these criteria must be satisfied to ensure a safe separation. For any pair of aircraft $\{1, 2\}$, these rules can be expressed by the following relationship:

$$d_H \geq sep_H \quad or \quad d_V \geq sep_V$$

where in cartesian coordinates $d_H = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$ represents the horizontal separation and $d_V = |z_2 - z_1|$ the vertical separation. Separation criteria impose lower-bounds on these distances:

$$sep_H = 3\ nm \qquad sep_V = 1000\ feet$$

The present approach to increase the operational capacity of the Air Traffic Control system consists of trying to reduce these values under specific circumstances.

The relationship above and the next three relationships, all expressed with first-order-logic notation, are equivalent:

$$\neg(d_H < sep_H \ \ and \ \ d_V < sep_V)$$

$$d_H < sep_H \ \Rightarrow \ d_V \geq sep_V \tag{6.1}$$

$$d_V < sep_V \ \Rightarrow \ d_H \geq sep_H$$

These relationships mean that horizontal and vertical separation must not be simultaneously violated. In the case of horizontal violation, vertical separation must be ensured for a safe separation; or alternatively in the case of vertical violation, horizontal separation must be ensured.

6.2.2 "Conflict-Free" Trajectories

A trajectory is defined as a set of points in a four-dimensional space. The coordinates of a point $M(t,x,y,z)$ are the temporal coordinate t, and the spatial cartesian coordinates x,y,z where z corresponds to the vertical component. Trajectory $\mathcal{T}$ is parameterized in t over an interval I_t:

$$\mathcal{T} \ = \ \{ \ M(t,x,y,z) \mid t \in I_t \ , \ \forall t \in I_t \ \begin{cases} x = x(t) \\ y = y(t) \\ z = z(t) \end{cases} \ \}$$

If at a given time $aircraft_1$ and $aircraft_2$ are respectively at points M_1 and M_2, they are said to be "Conflict-free" if the relationships 6.1 expressed above between the coordinates of M_1 and M_2 are satisfied.

Two trajectories $\mathcal{T}_1$ and $\mathcal{T}_2$ are said to be conflict-free (or kinematically conflict-free) if at any time along the trajectories the positions of the aircraft in space are such that the separation criteria are satisfied.

$$\forall M_1 \in \mathcal{T}_1, \ \forall M_2 \in \mathcal{T}_2, \ \ \textit{Conflict-free}(M_1, M_2)$$

or equivalently

$$\forall t \in I_{t_1} \cap I_{t_2}, \ \ \textit{Conflict-free}(M_1(t), M_2(t))$$

Two trajectories are said to be "geometrically conflict-free" if their geometric supports are such that the separation criteria expressed in terms of distance are

satisfied independently from the kinematic aspect of the motion (i.e. independently of the temporal law of motion).

$$\forall M_1 \in Support(\mathcal{T}_1),\ \forall M_2 \in Support(\mathcal{T}_2),\ \ Conflict - free(M_1, M_2)$$

or equivalently with $I = I_{t_1} \cap I_{t_2}$ and $\mathcal{I} = \{\varphi,\ \varphi : I \rightarrow I\ ,\ \varphi\ injective\}$

$$\forall (\varphi_1, \varphi_2) \in \mathcal{I} \times \mathcal{I},\ \forall t \in I,$$
$$Conflict - free(M_1(\varphi_1(t)), M_2(\varphi_2(t)))$$

Being geometrically conflict-free is a stronger property for two trajectories than being kinematically conflict-free. Two geometrically conflict-free trajectories are necessarily kinematically conflict-free. Obviously only the weaker of these properties is required by Separation Standards. The use of the two notions will be combined for an efficient conflict detection.

6.2.3 Air or Ground-Coordinates

Flight path plans are defined with reference to the ground but their representation combines both systems of reference. States include the position of the aircraft expressed in ground-coordinates but also air and ground-speed. Operators describe aircraft maneuvers. Their parameters are kinematic characteristics of the motion with reference to the air. The distance of separation between aircraft is an invariant which does not depend on the choice of a system of reference. Conflict detection may be performed alternatively in air or ground-coordinates with the main concern of efficiency.

The "geometric filtering" process which is described in 6.4 is entirely performed in ground-coordinates. It mainly uses the end-points of flight path legs and tries to infer that the legs are geometrically conflict-free. The position of the end-points are state variables defined in ground-coordinates.

The "kinematic detection" process which is described in 6.5 is performed in air-coordinates. At an initial time the origins of both systems of coordinates are matched. Trajectories with reference to the air are then entirely specified from these initial conditions by the equations of the aircraft motions which are implicit in the representation of the flight path plans. The "local geometric filtering" process which complements the "kinematic detection" uses the geometric shapes of flight path segments which are straight-lines and circles with reference to the air.

If the wind is uniform, all aircraft are influenced by the same wind and their relative velocity is alternatively the difference between the air-speed or ground-speed vectors. If the wind varies with position, the notion of global air-coordinates, in which the effect of wind could be ignored, disappears. However wind can still be considered as uniform on a local basis. Accuracy is required in the determination

of separations when aircraft are close enough that they may conflict. The necessary degree of locality concerns the average wind at the scale of the minimum distances imposed by separation criteria (three nautical miles horizontally and one thousand feet in altitude).

If any doubt remains, a "time-simulation" of the trajectories is available. Simulation is performed with reference to the ground. The method consists of computing several intermediate states and of testing each of them for conflict. The successive steps of conflict detection are described below.

6.3 Qualitative Filtering Using Properties of the Patterns

Flight paths built from the same pattern share common properties which are inherent to the design of the pattern. They have similar geometric shapes and involve the same sequencing of maneuvers. The conflict detection process can take advantage of these properties to infer that portions of the paths are conflict-free. This first filtering uses some qualitative reasoning based on common-sense and empirical rules derived from high-level properties of the patterns.

Flight paths leading to the same or parallel runways are built with respect to the same orientation, i.e. the direction of the runway centerlines. Many of their legs are consequently parallel or orthogonal. There exists some simple geometric properties about the relative positions of the legs which make it possible to infer that portions of the paths assigned to different aircraft can not conflict. When the paths lead to runways which are not parallel, simplicity disappears since the rules should take into account an arbitrary angle between the centerlines.

Consider, as an example, the case of two flight paths, one of type "arrival-trombone", and one of type "overhead-trombone", which lead aircraft to the same runway. Detailed descriptions of these two patterns appear in chapter 4. The first step consists of checking whether the plans overlap in time. If it is not the case, the paths are obviously conflict-free. Otherwise some of the following rules may be applied:

- Aircraft in different holding stacks are conflict-free. The "exit-from-holding-uniform-move" and the "turns-to-arrival-leg" segments are also conflict-free.

- It is the role of the stack management function to ensure that aircraft in the same holding stack are safely separated. Two aircraft in the same holding stack are considered to be conflict-free if they are not at the same altitude during overlapping time-intervals. That implies that two aircraft are not allowed to fly the same race-track loop at the same time and altitude.

- If the aircraft arrive from opposite sides of the runway, then for our assumed pair of arrival patterns, the trombone parts of the patterns are performed on the same side:
  - The "arrival leg" and the "turn to downwind" of the path of type "arrival-trombone" and the "arrival leg" and the "turn to downwind" of the path of type "overhead-trombone" can not conflict since the downwind leg of the former is more than four nautical miles away from the runway centerline. This four-nautical mile distance corresponds to the minimum quantized value of the downwind offset according to the terminal area organization presented in 4.3.2 and figure 4.2.
  - If the "downwind lateral offsets" have different values, then the downwind legs are conflict-free. Because of the quantization of this degree of freedom with a three nautical mile increment, the values differ by at least the minimum horizontal separation.
  - If the "downwind lateral offset" of the "arrival-trombone" path is strictly greater than the "downwind lateral offset" of the "overhead-trombone" path, the "divergent leg" of the latter path is conflict-free.
- If the two aircraft arrive from the same side of the runway the trombone parts of the patterns are performed on opposite sides:
  - The "turn to base" of the "arrival-trombone" path is conflict-free from the other path. Similarly the "overhead turn", the "divergent leg", the "turn to downwind", the "downwind leg", and the "turn to base" of the "overhead-trombone" path are conflict-free.
  - Note: assuming "intercept legs" of 30 seconds with no deceleration phase, "final-approach-speeds" of 100 knots, angular speeds of 3 degree per second and angles of interception of 30 degrees, the lateral distance between the beginning of the turns to the "intercept legs" is 1.9 nautical mile. This is not enough to ensure a minimum horizontal separation between the "base legs".
  - If the paths start from the same holding point and if the angle between the straight-line orthogonal to the runway centerline and the "arrival leg" of the "arrival-trombone" path is greater than the corresponding angle of the "overhead-trombone" path, i.e. the "arrival leg" of the latter does not intercept the "downwind leg" of the former, this "downwind leg" is conflict-free.

Parts of the paths which were not ruled out as conflict-free in this example require a more detailed detection. Nevertheless it could be inferred by simple geometric reasoning that most of the segments can not conflict. More complex

rules involving timing considerations may be devised but it is probably not worthwhile spending too much effort on the design of intricate rules which would be applicable only in specific situations. If many nested conditionals are needed to apply a sophisticated rule, it is likely that a systematic detection at a deeper level of abstraction would be as fast for more detailed results. The design and the implementation of a large set of such rules would involve the same difficulties of definition and maintenance as for a knowledge-based system. In the state of the art, the field of Artificial Intelligence has not provided yet a general framework for qualitative spatio-temporal reasoning which would allow primarily qualitative detection.

These rules are pattern-dependent. If the whole conflict detection process were based on such rules the system would become obsolete as soon as patterns are altered. Patterns however are supposed to be interchangeable. Their definition should evolve to follow the changes in *ATC* procedures and aircraft technology. They are treated as data on which the machineries for path generation and conflict detection do not depend. Some specific knowledge about the patterns allows a speed-up of the conflict detection process by an early filtering of portions of the paths which are obviously conflict-free using high-level properties of the patterns. The underlying conflict detection logic relies on the representation of the paths, and works independently from this qualitative filtering process.

6.4 Geometric Filtering

Qualitative reasoning relying on high-level properties of the patterns makes it possible to rule out those parts of the paths which do not conflict as a consequence of the shape of the patterns. A geometric filtering process is now applied to the rest of the paths at intermediate levels of abstraction. The goal of this step of geometric filtering is not completeness. It is not intended to detect all the existing conflicts, but only to reduce the extent of more detailed detection when it is likely that entire portions of the paths are not in conflict.

6.4.1 Principle of Geometric Filtering

The idea of geometric filtering consists of surrounding the trajectories which are to be tested for conflict with control volumes which are defined in the four dimensional spatio-temporal space; if these volumes do not intercept and even are at a sufficient distance one from the other, it is possible to infer that the portions of trajectory they contain are conflict-free.

A trajectory is first broken into parts which have a geometric entity, such as a path leg which is supported by a straight-line. Each part is surrounded by a control volume which entirely contains it and occupies as little space as possible to reduce the probability of interaction with other trajectories so as to gain in

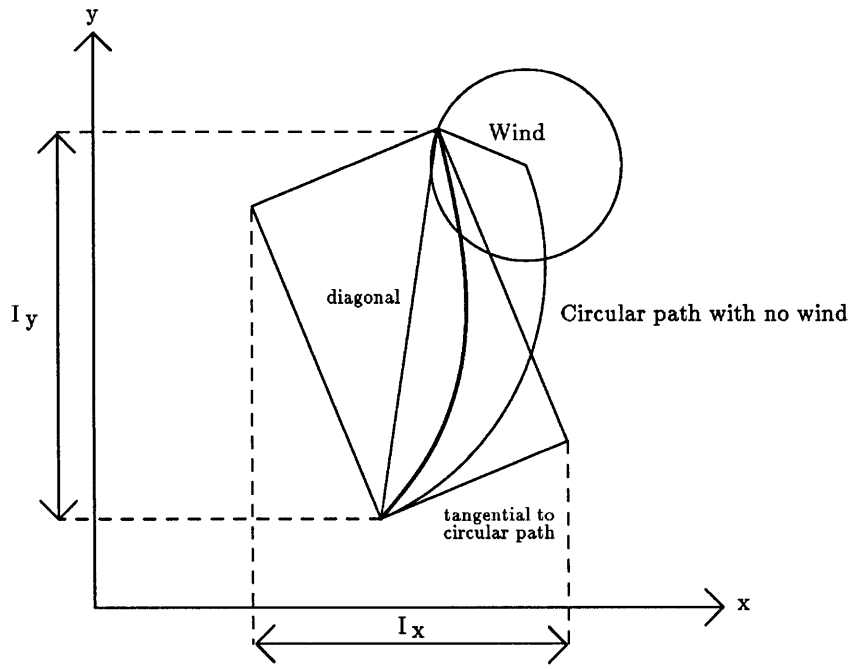

Figure 6.1: Control Volume for an Arc of Cycloidal Curve

efficiency of the filtering. Control volumes should have easily defined boundaries, and testing for the intersection of control volumes or determining the distances between them should be computationally inexpensive.

6.4.2 Implementation Choices for Control Volumes

The portions of trajectory which have to be surrounded by control volumes correspond to flight phases according to the hierarchical representation of the flight paths presented in chapter 3. They span over an entire leg which is composed of several segments, or only a turn. Each portion of trajectory has a simple geometric shape.

Control volumes are hypercubes whose faces are parallel to the directing planes of the cartesian space.

$$CV = I_t \times I_x \times I_y \times I_z$$

where I_i's for $i \in \{x, y, z, t\}$, are the component intervals in each of the dimension. The minimum control volume of the chosen shape (minimum in the sense of inclusion) are such that the component intervals are the projections of the portion of trajectory on the coordinate axis. Since spatial coordinates are continuous functions of time each interval of the cartesian product is a convex interval.

In the case of a straight-line portion of trajectory, the associated control volume is entirely defined by the end-points and the time-interval during which the trajectory is flown. These data are contained in the states at the extremities of the segment, whatever the law of motion between the states might be. In the case of an arc of circle some of the parameters of the transformation are necessary in addition to the extremal states to characterize the orientation and curvature of the arc. Bounds of the projection interval have a simple analytic expression which involves little computation. When complex shapes are introduced, such as arcs of cycloidal curves (to consider the vectorial component of the wind combined with a turn), precise bounds may not be easily accessible. Choosing a larger interval may be perfectly acceptable since that only relaxes the minimality condition.

The definition of a control volume in the slightly trickier case of an arc of cycloidal curve is now presented. The vertices of a rectangle containing the curve are first determined. Then the rectangle is projected on the coordinate axis. The rectangle is chosen such that the end-points of the cycloidal curve belong to one of its diagonals. An additional condition is that an edge of the rectangle containing the end-point corresponding to the initial state is tangent to the arc of circle which would have been obtained in the absence of wind. The integrated effect of the wind can be represented by the product of the wind speed vector by the duration of the turn. The final state can be easily obtained by adding this vector to the position of the end-point of the circle which would have been obtained in the absence of wind. Figure 6.1 shows the construction of the control volume which is entirely defined by these conditions.

The efficiency of the geometric filtering is actually a compromise between the difficulty to define the control volumes and their sizes on which depends the probability of being closer than a minimum distance from other control volumes.

6.4.3 Logic for Geometric Filtering

Trajectories intersect when they have a common point in time and space. Consequently a necessary condition for trajectories to intersect is that the respective control volumes intersect. Similarly a necessary condition for trajectories to be conflicting is that the respective control volumes are conflicting, in the sense that the minimal distances between the volumes are not satisfied.

The distance between two intervals I_1 and I_2 is the minimal distance between any two points x_1 and x_2 of I_1 and I_2 respectively:

$$D(I_1, I_2) \;=\; min_{\substack{x_1 \in I_1 \\ x_2 \in I_2}} \; d(x_1, x_2)$$

Minimal separation between control volumes ensures minimal separation between the corresponding trajectories.

The logic of the filtering process using the control volumes described above is the following:

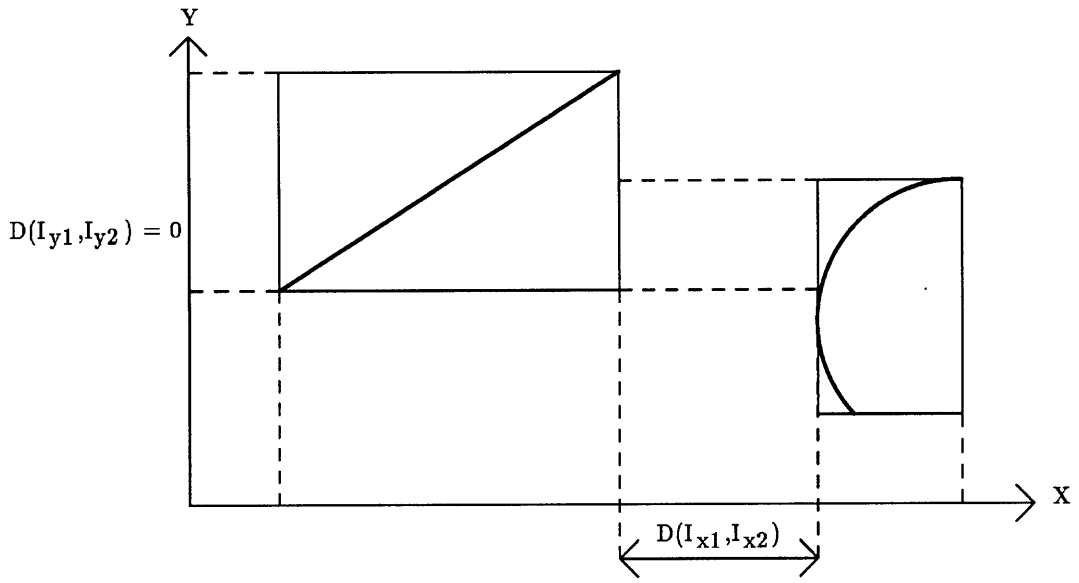

Figure 6.2: Horizontal Geometric Filtering

1. Two trajectories may conflict only if they are flown during overlapping time-intervals:

$$D_T = D(I_{t1}, I_{t2}) > 0 \Rightarrow \textit{Conflict-free}(\mathcal{T}_1, \mathcal{T}_2)$$

2. Two trajectories may conflict only if vertical separation between the control volumes is violated:

$$D_V = D(I_{z1}, I_{z2}) \geq sep_V \Rightarrow \textit{Conflict-free}(\mathcal{T}_1, \mathcal{T}_2)$$

3. Two trajectories may conflict only if horizontal separation between the control volumes is violated. Figure 6.2 illustrates this case. The minimum distance D_H between the two rectangles can be simply expressed in terms of the distances between the projection intervals on each of the two axis.

$$D_H^2 = D(I_{x1}, I_{x2})^2 + D(I_{y1}, I_{y2})^2 \geq sep_H^2 \Rightarrow \textit{Conflict-free}(\mathcal{T}_1, \mathcal{T}_2)$$

4. Otherwise a more detailed detection is necessary to determine whether the trajectories are actually conflict-free:

$$\textit{Potentially-conflicting}(\mathcal{T}_1, \mathcal{T}_2)$$

Note 1: The following property is a consequence of the continuity of the trajectories:

$$I_{x1} \subset I_{x2} \text{ and } I_{y2} \subset I_{y1} \Rightarrow \textit{Geometrically-conflicting}(\mathcal{T}_1, \mathcal{T}_2)$$

Note 2: If any of the component intervals is reduced to a point, the corresponding control volume is degenerated, but the method still holds.

Geometric filtering ensures that trajectories flown during overlapping time intervals are geometrically conflict-free. The portions of the trajectories which are still potentially conflicting have to be subsequently tested.

6.5 Kinematic Detection

The kinematic detection operates on flight path segments at the level of abstraction of the State Space representation and relies on an algebraic determination of the minimum distance between portions of trajectories. This step of the "cascaded" filtering process is complete, in the sense that it can detect all existing conflicts.

A segment corresponds to the application of an operator between two states. The transformation between the two states is entirely characterized by the parameters of the operator. This symbolic representation of the trajectories is exploited by a mathematical conflict detection. The detection is based on equations of kinematics and geometry which are solved algebraically. All the cases of interaction for all possible pair of segments are considered, requiring many specific algorithms which are the concisely programmed results of a few pages of algebra. The law of motion and the geometry of the portions of trajectory corresponding to each segment of a potentially conflicting pair are used to determine the expression of the minimal distance between the segments. This distance is compared to the separation criteria imposed by Separation Standards to conclude whether the segments are conflicting or not.

6.5.1 Matching the Overlapping Time Intervals

Trajectories are parametrized in t considering that time is monotonically increasing. At every level of abstraction an element of trajectory is composed of a sequence of subelements ordered with respect to time. Consider a portion of trajectory $\mathcal{T}$ which is composed of s segments S_1 through S_s. The time interval corresponding to $\mathcal{T}$ is a convex interval composed of juxtaposed time intervals in increasing order:

$$\mathcal{T} = [\mathcal{S}_1, ..., \mathcal{S}_s] \qquad I_t(\mathcal{T}) = \cup_{i=1}^{s} I_t(\mathcal{S}_i)$$

Time monotonicity is explicit in the representation of the trajectories:

$$i < j \Rightarrow I_t(S_i) \leq I_t(S_j)$$

(in the sense that $\forall t_i \in I_t(\mathcal{S}_i),\ \forall t_j \in I_t(\mathcal{S}_j),\ t_i \leq t_j$)

Consider two portions of trajectories $\mathcal{T}_1$ and $\mathcal{T}_2$ which have to be tested for conflict. The first step consists of determining the set of time intervals of $I_t(\mathcal{T}_1)$ and $I_t(\mathcal{T}_2)$ which overlap. The procedure which matches the overlapping time intervals returns a set of couples, where the first element is the index of a segment of $\mathcal{T}_1$ and the second element the index of a segment of $\mathcal{T}_2$; and for each couple, a time interval of time overlap which is the intersection of the time intervals of the two segments. Further investigation focuses on this set of potential conflicts between two path segments over the corresponding time intervals of time overlap.

6.5.2 Altitude Overlap

The following equations consider the cases of climb or descent with a constant rate, or constant altitude. More complex altitude profiles may be decomposed into a succession of linear phases with a constant rate. However, vertical conformance of aircraft to planned paths is not very precise in current ATC operations .

Let S_1 and S_2 be two segments which overlap in time,
and $I_t = I_t(S_1) \cap I_t(S_2)$, the interval of time overlap.

$$\begin{cases} \forall t \in I_t(S_1), & z_1 = z_1^0 + r_1 t \\ \forall t \in I_t(S_2), & z_2 = z_2^0 + r_2 t \end{cases}$$

The expression of the altitude separation is:

$$\Delta z = \mid z_2^0 - z_1^0 + (r_2 - r_1)t \mid$$

Let $\Delta r = |r_2 - r_1|$.

- Case $\Delta r = 0$:

  The altitude separation remains constant. Then, if $\Delta z \geq sep_V$ at either of the bounds of I_t, vertical separation is satisfied and the segments are conflict-free over I_t . Otherwise let the interval of time and altitude overlap be $I_{tz} = I_t$

- Case $\Delta r \neq 0$:

  Then, $\Delta z(t^0) = 0$ when $t^0 = -\frac{z_2 - z_1}{r_2 - r_1}$

  $\Delta z(t_z^{\pm}) = sep_V$ when $t_z^{\pm} = t^0 \pm \frac{sep_V}{\Delta r}$

  The condition of violation of the vertical separation criterion is the following:

$$\Delta z < sep_V \iff t \in I_t \cap I_z \quad , \quad I_z =]t_z^- \ t_z^+[$$

  Let the time interval of time and altitude overlap be $I_{tz} = I_t \cap I_z$.

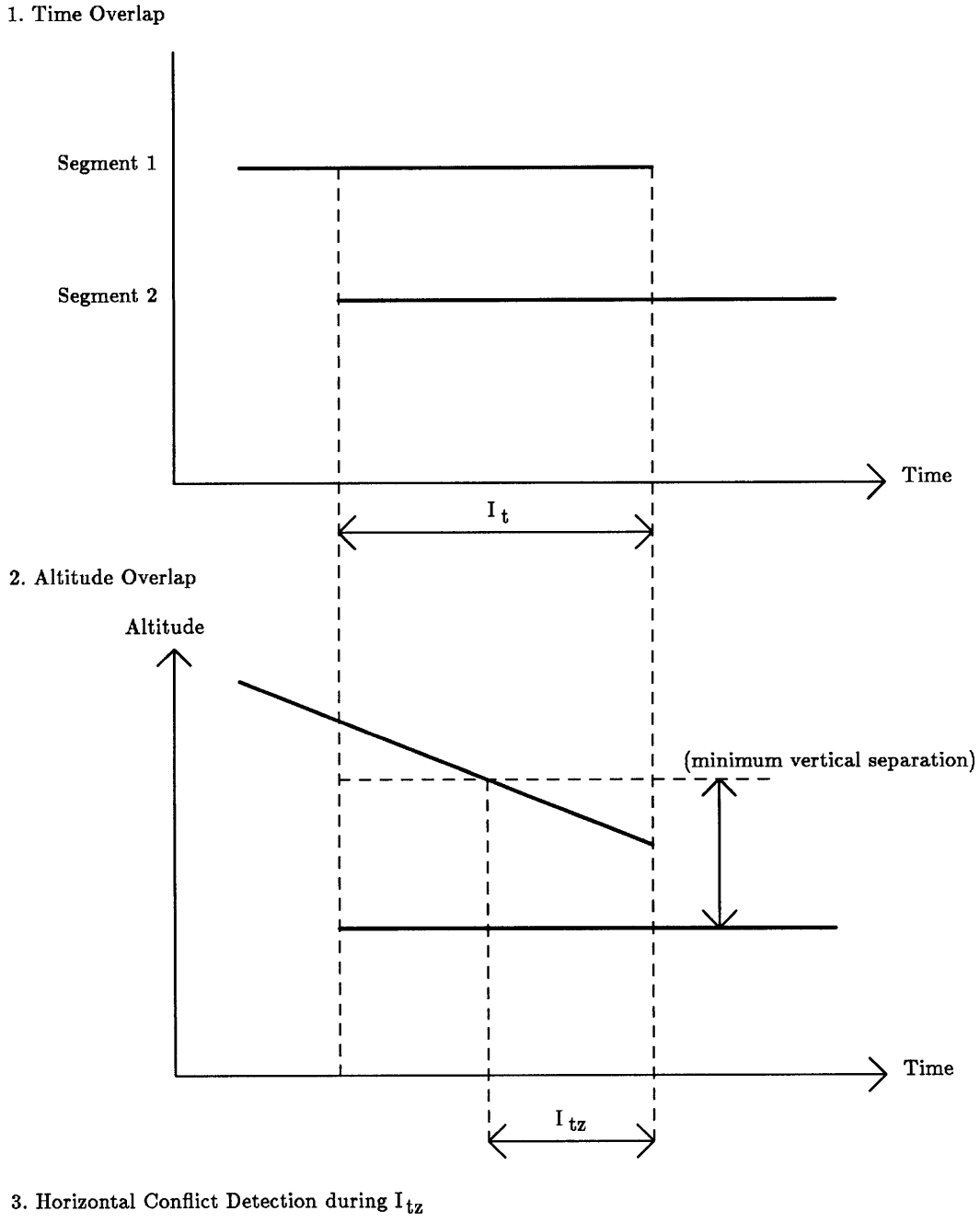

Figure 6.3: Time and Altitude Overlap

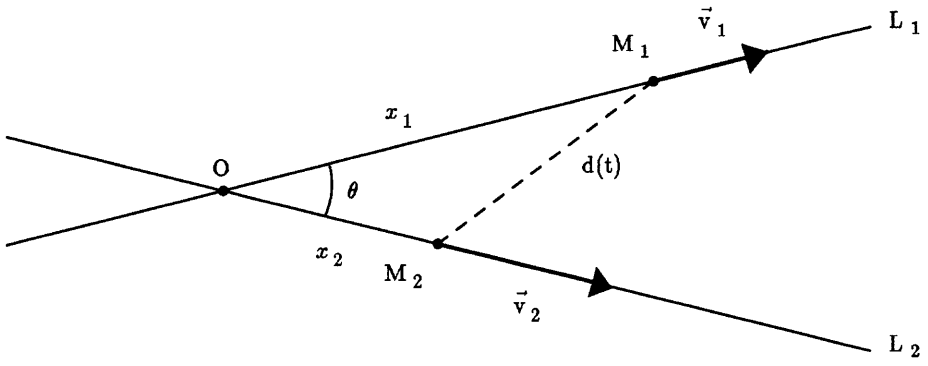

Figure 6.4: Separation between Two Uniform-Move Segments

Figure 6.3 summarizes the successive steps of kinematic conflict detection involving the determination of I_{tz} . Further investigations focus on the set of potential horizontal conflicts between path segments over the corresponding time intervals of time and altitude overlap. For simplicity of notation the time interval I_{tz} will be noted I.

6.5.3 Horizontal Conflict Detection

Analytic expressions of the horizontal separation between two segments are used to determine the minimum separation over the time interval $I = [t_i \; t_f]$. This minimum separation has to be compared with the minimum horizontal separation imposed by Separation Standards. Several cases are to be considered depending on the geometry of the trajectory segments as well as on the temporal laws of motion.

The most frequent case is certainly the one when two aircraft fly in a straight-line at a constant speed. This corresponds to the interaction between two uniform-move segments. The algebra for this simple case is presented below as an example of the analytic approach.

Uniform Move / Uniform Move

Aircraft fly along the straight-lines L_1, L_2 with constant speeds. Consider the general case of converging lines. L_1, L_2 of director vectors $\vec{u_1}$, $\vec{u_2}$ intersect at point O. In the case of uniform moves where both air and ground-speed remain constant, the equations are the same in both systems of reference (v denotes air or ground-speed).

$$O\vec{M}_1 = x_1\vec{u_1} \;\; , \;\; O\vec{M}_2 = x_2\vec{u_2} \;\; , \;\; \theta = |\; \widehat{\vec{u_1}\;\vec{u_2}} \;|$$

$$\begin{cases} x_1 = x_1^0 + v_1 t \\ x_2 = x_2^0 + v_2 t \end{cases}$$

$$\begin{aligned} d^2(t) &= d^2(M_1(t), M_2(t)) \\ &= (O\vec{M}_1)^2 + (O\vec{M}_2)^2 - 2(O\vec{M}_1.O\vec{M}_2) \\ &= x_1^2 + x_2^2 - 2x_1x_2cos\theta \\ &= (x_1^0 + v_1t)^2 + (x_2^0 + v_2t)^2 - 2(x_1^0 + v_1t)(x_2^0 + v_2t) \end{aligned}$$

$\vec{v_1} \neq \vec{v_2}$ since the straight-lines are not parallel
$d^2(t), t \in R$, is a quartic which admits a minimum of value d^{*2} for $t = t^*$ with:

$$t^* = \frac{x_1^0(v_2cos\theta - v_1) + x_2^0(v_1cos\theta - v_2)}{v_1^2 + v_2^2 - 2v_1v_2cos\theta}$$

$$d^* = \frac{\mid x_1^0v_2 - x_2^0v_1 \mid sin\theta}{\sqrt{v_1^2 + v_2^2 - 2v_1v_2cos\theta}}$$

Depending on the relative positions of t^* and I, the minimum separation between the trajectories during the time interval I is :

$$\begin{cases} & t^* < t_i & min_{\ t \in I}\ d(t) = d(t_i) \\ t_i < & t^* < t_f & min_{\ t \in I}\ d(t) = d^* \\ t_f < & t^* & min_{\ t \in I}\ d(t) = d(t_f) \end{cases}$$

Introduce in the expressions of t^* and d^*, the times t_1^0, t_2^0 at which the aircraft may cross the intersection point O, and the time difference Δt :

$$\begin{cases} x_1 = v_1(t_1 - t_1^0) \\ x_2 = v_2(t_2 - t_2^0) \end{cases}$$

$$t^* = \frac{t_1^0v_1^2 + t_2^0v_2^2 - (t_1^0 + t_2^0)v_1v_2cos\theta}{v_1^2 + v_2^2 - 2v_1v_2cos\theta}$$

$$d^* = \frac{v_1v_2sin\theta \mid t_1^0 - t_2^0 \mid}{\sqrt{v_1^2 + v_2^2 - 2v_1v_2cos\theta}} = \frac{\| \vec{v_1} \wedge \vec{v_2} \|}{\| \vec{v_1} - \vec{v_2} \|}\Delta t$$

Geometry of the Closest Approach in the Case of Uniform Moves:

Assume without loss of generality that aircraft number 1 crosses the intersection point before aircraft number 2, i.e. $t_1^0 < t_2^0$ .There are three cases:

- $t^* < t_1^0 < t_2^0$

  The closest approach occurs while both aircraft are converging towards the intersection. The following relationship between the aircraft speeds is obtained by substituting t^* by its expression above in the inequality:

  $$t^* < t_1^0 \iff v_2 < v_1 cos\theta \ \ (< v_1)$$

  The faster aircraft crosses the intersection first in this case.

- $t_1^0 \leq t^* \leq t_2^0$

  The closest approach occurs while the first aircraft has already crossed the intersection and is diverging from it, whereas the second aircraft is converging to it.

  $$t_1^0 \leq t^* \leq t_2^0 \iff \left\{ \begin{array}{lcl} v_1 cos\theta & \leq & v_2 \\ v_2 cos\theta & \leq & v_1 \end{array} \right.$$

  - if $\theta \in [0\ \frac{\pi}{2}]$

    Let r denote the ratio of the speeds $\frac{v_2}{v_1}$ or $\frac{v_2}{v_1}$. The relationship above between the times implies $r \in [cos\theta\ \frac{1}{cos\theta}]$.

  - if $\theta \in [\frac{\pi}{2}\ \pi]$ no relationship between the speeds is implied.

- $t_1^0 \leq t_2^0 \leq t^*$

  The closest approach occurs while both aircraft are diverging from the intersection.

  $$t_2^0 < t^* \iff v_1 < v_2 cos\theta \ \ (< v_2)$$

  The slower aircraft crosses the intersection first in this case.

Other Cases

The cases which have been treated analytically and for which a simple expression of the minimum separation is available, are the following:

- uniform move / uniform move

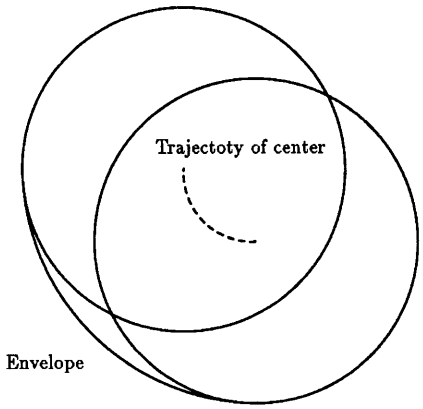

Figure 6.5: Swept Area during a Turn

- uniform move / acceleration
- acceleration / acceleration
- turn / turn (with the same angular speed)

Note 1: The horizontal projection of a climb or descent phase with constant rate when the aircraft flies in straight-line is a uniform move.

Note 2: Since the rate of turn is standardized to the usual value of three degrees per second, the cases $\omega_1 = \pm\omega_2 = \omega_{standard}$ for which an analytic solution could be provided are likely to be applicable most of the time. As with vertical rates, we cannot be certain about the conformance of aircraft to turn rates.

All these cases can be reduced to the analysis of polynomial equations, quartic, cubic, or quadric. The following cases involve more complex trigonometric equations or the combinations of polynomial and trigonometric equations:

- uniform move / turn
- acceleration / turn
- turn / turn (with arbitrary angular speeds)

When the algebra becomes untractable, a time-simulation of the trajectories during the time interval I is possible. To avoid the problems imputable to numerical mathematical methods, an alternative analytic method of "local geometric detection" was given priority.

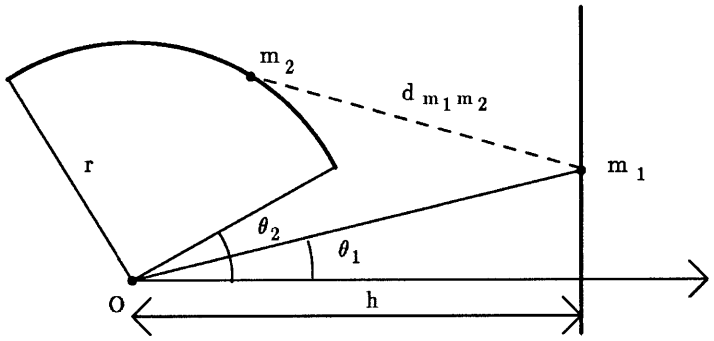

Figure 6.6: Distance between a Straight-Line Segment and an Arc of Circle

Local Geometric Detection

Instead of directly proving that two segments are kinematically conflict-free, we try to show that they are geometrically conflict-free. Considering that it is a stronger property it may be the case that the former is satisfied while the latter is not. Time-simulation is then the last recourse. However, some qualitative remarks suggest that it is "not very likely" to occur. Consider the area swept by a circle whose center moves along a trajectory segment. This circle corresponds to the horizontal boundary of the protected air-space surrounding each aircraft with respect to Separation Standards. Kinematic separation requires that the circles surrounding any pair of aircraft do not intercept at any time. Geometric separation requires that the complete areas swept by the circles do not intercept during the corresponding time interval. When the length of the trajectory segment is smaller than the radius of the circle, the instantaneous area or the total swept area do not differ much. Also, a turn is compact in the sense that it does not require much space and the area swept by a circle whose center moves along a circular arc (figure 6.5) is smaller than in the case of a straight-line motion of same duration.

The geometric cases which have to be treated and for which the necessary algebra is presented below are the following:

- straight-line segment / arc of circle
- arc of circle / arc of circle

Now it is possible to show how the notion of geometric separation is handled analytically and to give an idea of the complexity of the resulting formulas. The reader who is not interested in the detail of the equations may skip to page 141.

Straight-Line Segment / Arc of Circle:

Let L be a straight-line, and C a circle of center O and radius r, as depicted in figure 6.6. h denotes the distance between O and L. Polar coordinates are used, of origin O and axis orthogonal to L, oriented from O to L. Let S_1 be a segment of L defined by its end-points of argument Θ_1, Θ_1'. Let S_2 be an arc of C limited by the angles Θ_2, Θ_2'.

With the notation:

$$\begin{array}{ll} m_1 \in S_1 & arg(m_1) = \theta_1 \in [\Theta_1\ \Theta_1'] \subset]-\frac{\pi}{2}\ \frac{\pi}{2}[\\ m_2 \in S_2 & arg(m_2) = \theta_2 \in [\Theta_2\ \Theta_2'] \end{array}$$

the expression of the distance between two arbitrary points m_1 and m_2 of S_1 and S_2 is:

$$\begin{aligned} d^2_{m_1m_2} &= (\vec{Om_1})^2 + (\vec{Om_2})^2 - 2(\vec{Om_1}.\vec{Om_2}) \\ &= (\frac{h}{cos\theta_1})^2 + r^2 - 2r\frac{h}{cos\theta_1}cos(\theta_2 - \theta_1) \end{aligned}$$

Using the non-dimensional parameter $\eta = \frac{h}{r}$ and with $\mathcal{V} = [\Theta_1\ \Theta_1'] \times [\Theta_2\ \Theta_2']$:

$$\delta^2_{m_1m_2} = (\frac{\eta}{cos\theta_1})^2 - 2\eta\ \frac{cos(\theta_2 - \theta_1)}{cos\theta_1} + 1$$

$$d_{S_1S_2} = \min_{\substack{m_1 \in S_1 \\ m_2 \in S_2}} d_{m_1m_2} = r.\min_{\mathcal{V}}\ \delta_{m_1m_2}(\theta_1, \theta_2)$$

The function $\delta_{m_1m_2}(\theta_1, \theta_2)$ is continuous on the closed set $\mathcal{V}$. The minimum $d_{S_1S_2}$ is reached either in the interior of $\mathcal{V}$ at points where the differential is equal to zero or on the boundary of $\mathcal{V}$.

For θ_1 fixed, $\delta^2_{m_1m_2}$ admits minima of value δ^{*2} for $\theta_2 = \theta_1\ (2\pi)$:

$$\delta^{*2}(\theta_1) = (1 - \frac{\eta}{cos\theta_1})^2$$

Two cases have to be considered depending on the value of η:

- $\eta \leq 1$:

  $\theta_1 = \theta_2 = \pm Acos\ \eta\ (2\pi)$ correspond to minima of value nul for which the straight-line and the circle intersect.

  $\theta_1 = \theta_2 = 0$ or $\pi\ (2\pi)$ correspond to maxima.

$$\min_{R \times R} d_{m_1m_2} = 0$$

- $\eta > 1$

  The function $f(x) = (1 - \frac{\eta}{x})^2$ admits a minimum of value $(1-\eta)^2$ on the interval $[-1\ 1]$ for $x = 1$.

  $\theta_1 = \theta_2 = 0\ (2\pi)$ correspond to minima of $\delta_{m_1 m_2}$ of value $\eta - 1$.

  $\theta_1 = \theta_2 = \pi\ (2\pi)$ correspond to maxima.

$$min_{\ R \times R}\ d_{m_1 m_2} \ = \ h - r$$

The minima corresponding to a zero differential are absolute minima. If they do not belong to $\mathcal{V}$, the minimum of $\delta_{m_1 m_2}$ is reached on the boundary of $\mathcal{V}$.

$$Boundary(\mathcal{V}) = \cup_{i \in \{a,b,c,d\}}\ \mathcal{B}_i$$

- a) $\mathcal{B}_a \ = \ \{(\theta_1, \theta_2) \in R^2,\ \theta_1 = \Theta_1\ ,\ \theta_2 \in [\Theta_2\ \Theta_2']\ \}$

  $\delta_a(\theta_2) \ = \ (\frac{\eta}{cos\Theta_1})^2 \ - \ 2\eta \frac{cos(\theta_2 - \Theta_1)}{cos\Theta_1} \ + \ 1$

  Minima of value δ_a^* are reached for $\theta_{2\,a}^* = \Theta_1\ (2\pi)$:

$$\delta_a^* \ = \ |\ 1 \ - \ \frac{\eta}{cos\Theta_1}\ |$$

$$\left\{ \begin{array}{rcll} & \Theta_1 & < \Theta_2 & min_{\mathcal{B}_a}\delta_a \ = \ \delta_a(\Theta_2) \\ \Theta_2 \leq & \Theta_1 & \leq \Theta_2' & min_{\mathcal{B}_a}\delta_a \ = \ \delta_a^* \\ \Theta_2' < & \Theta_1 & & min_{\mathcal{B}_a}\delta_a \ = \ \delta_a(\Theta_2') \end{array} \right.$$

- b) $\mathcal{B}_b \ = \ \{(\theta_1, \theta_2) \in R^2,\ \theta_1 = \Theta_1'\ ,\ \theta_2 \in [\Theta_2\ \Theta_2']\ \}$

  The result is similar to the one obtained for $\mathcal{B}_a$ changing Θ_1 to Θ_1'.

- c) $\mathcal{B}_c \ = \ \{(\theta_1, \theta_2) \in R^2,\ \theta_1 \in [\Theta_1\ \Theta_1']\ ,\ \theta_2 = \Theta_2\}$

$$\begin{array}{rcccccc} \delta_c^2(\theta_1) & = & \eta^2(1 + \tan^2\theta_1) & - & 2\eta(cos\Theta_2 + sin\Theta_2\ \tan\theta_1) & + & 1 \\ \delta_c^2(t) & = & \eta^2(1 + t_1^2) & - & 2\eta(cos\Theta_2 + sin\Theta_2\ t_1) & + & 1 \end{array}$$

  with $t = \tan\theta_1\ ,\ t \in [T_1\ T_1']$.

  $\delta_c^2(t_1),\ t_1 \in R$, is a parabola which admits a minimum δ_c^{*2} for t_1^* such that $\eta t_1^* = sin\Theta_2$:

$$\delta_c^* \ = \ \sqrt{1 + \eta^2 - 2\eta cos\Theta_2 - sin^2\Theta_2} \ = \ |\ \eta - cos\Theta_2\ |$$

$$\left\{ \begin{array}{rcll} & t_1^* & < T_1 & min_{\mathcal{B}_c}\delta_c \ = \ \delta_c(T_1) \\ T_1 \leq & t_1^* & \leq T_1' & min_{\mathcal{B}_c}\delta_c \ = \ \delta_c^* \\ T_1' < & t_1^* & & min_{\mathcal{B}_c}\delta_c \ = \ \delta_c(T_1') \end{array} \right.$$

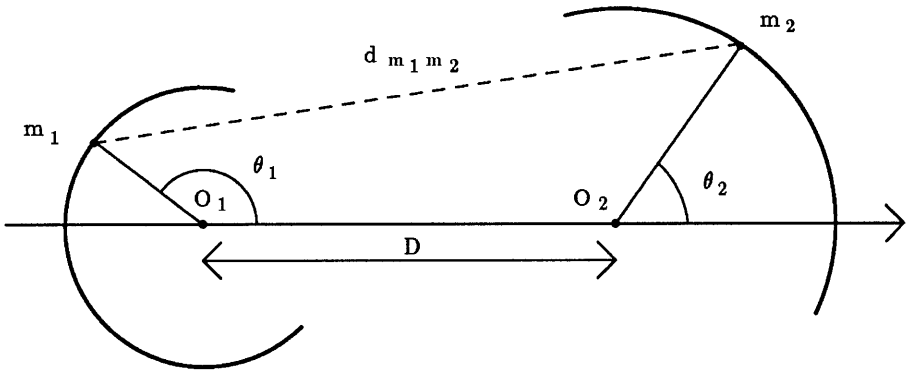

Figure 6.7: Distance between Two Arcs of Circle

- d) $\mathcal{B}_d = \{(\theta_1, \theta_2) \in R^2,\ \theta_1 \in [\Theta_1\ \Theta_1']\ ,\ \theta_2 = \Theta_2\}$
  The result is similar to the one obtained for $\mathcal{B}_c$ changing Θ_2 to Θ_2'.

$$min_{\ Boundary(\mathcal{V})}\ \delta_{m_1m_2} = inf_{\ i \in \{a,b,c,d\}}\ (min_{\ \mathcal{B}_i}\ \delta_i)$$

Arc of Circle / Arc of Circle:

Let C_i, for $i \in \{1,2\}$, be a circle of center O_i and radius R_i. D denotes the distance between the centers, as depicted in figure 6.7. S_i is a segment of C_i limited by the angles Θ_i, Θ_i' which are measured from the axis joining the centers, oriented from O_1 to O_2. $d_{m_1m_2}$ is the distance between two arbitrary points m_1, m_2 of the segments S_1, S_2 of argument θ_1, θ_2.

$$\begin{aligned} d_{m_1m_2} &= (\vec{m_1O_1})^2 + (\vec{O_1O_2})^2 + (\vec{O_2m_2})^2 \\ &+ 2(\ \vec{m_1O_1}.\vec{O_1O_2} + \vec{m_1O_1}.\vec{O_2m_2} + \vec{O_1O_2}.\vec{O_2m_2}\ ) \\ &= R_1^2 + D_2 + R_2^2 \\ &+ 2(-R_1Dcos\theta_1 - R_1R_2cos(\theta_2 - \theta_1) + R_2Dcos\theta_2\ ) \end{aligned}$$

Using the non-dimensional parameters $r_i = \frac{R_i}{D}$, $\delta_{m_1m_2} = \frac{d_{m_1m_2}}{D}$ and with $\mathcal{W} = [\Theta_1\ \Theta_1'] \times [\Theta_2\ \Theta_2']$:

$$\delta^2_{m_1m_2} = 1 + r_1^2 + r_2^2 - 2((r_1cos\theta_1 - r_2cos\theta_2) + r_1r_2cos(\theta_2 - \theta_1))$$

$$d_{S_1S_2} = min_{\substack{m_1 \in S_1 \\ m_2 \in S_2}}\ d_{m_1m_2} = D.\ min_{\ \mathcal{W}}\ \delta_{m_1m_2}(\theta_1, \theta_2)$$

The function $\delta_{m_1 m_2}(\theta_1, \theta_2)$ is continuous on the closed set $\mathcal{W}$. The minimum $d_{S_1 S_2}$ is reached either in the interior of $\mathcal{W}$ at points where the differential is equal to zero, or on the boundary of $\mathcal{W}$.

$$\begin{cases} p = \frac{\partial \delta^2_{m_1 m_2}}{\partial \theta_1}(\theta_1, \theta_2) &= 2r_1(sin\theta_1 - r_2 sin(\theta_2 - \theta_1)) \\ q = \frac{\partial \delta^2_{m_1 m_2}}{\partial \theta_2}(\theta_1, \theta_2) &= 2r_2(-sin\theta_2 + r_1 sin(\theta_2 - \theta_1)) \end{cases}$$

$$\begin{cases} p = 0 \\ q = 0 \end{cases} \iff r_1 sin\theta_1 = r_2 sin\theta_2 = r_1 r_2 sin(\theta_2 - \theta_1)$$

$$\iff \begin{cases} r_1 sin\theta_1 = r_2 sin\theta_2 \\ (r_1 sin\theta_1)(1 - r_1 cos\theta_1 + r_2 cos\theta_2) = 0 \end{cases}$$

$$\iff \begin{cases} \theta_1 = 0\ (\pi) \\ \theta_2 = 0\ (\pi) \end{cases} \quad or \quad \begin{cases} r_1 sin\theta_1 = r_2 sin\theta_2 \\ 1 - r_1 cos\theta_1 + r_2 cos\theta_2 = 0 \end{cases}$$

$$\begin{cases} r = \frac{\partial^2 \delta^2_{m_1 m_2}}{\partial \theta_1^2}(\theta_1, \theta_2) &= 2r_1(cos\theta_1 + r_2 cos(\theta_2 - \theta_1)) \\ s = \frac{\partial^2 \delta^2_{m_1 m_2}}{\partial \theta_1 \partial \theta_2}(\theta_1, \theta_2) &= -2r_1 r_2 cos(\theta_2 - \theta_1)) \\ t = \frac{\partial^2 \delta^2_{m_1 m_2}}{\partial \theta_2^2}(\theta_1, \theta_2) &= 2r_2(-cos\theta_1 + r_1 cos(\theta_2 - \theta_1)) \end{cases}$$

Let $\Delta = s^2 - rt$. Among the points where the differential is equal to zero, points where Δ is negative are extrema. r, t are positive at minima and negative at maxima. Points where Δ is positive are saddle points.

Point	$\theta_1(2\pi)$	$\theta_2(2\pi)$	$\frac{\Delta}{4r_1 r_2}$	$\frac{r}{r_1}$	$\frac{t}{r_2}$
1	0	0	$1 - r_1 + r_2$	$1 + r_2 > 0$	$-1 + r_1$
2	π	π	$1 + r_1 - r_2$	$-1 + r_2$	$1 + r_1 > 0$
3	0	π	$-1 + r_1 + r_2$	$1 - r_2$	$1 - r_1$
4	π	0	$-1 - r_1 - r_2 < 0$	$-1 - r_2 < 0$	$-1 - r_1 < 0$

Point 4 always corresponds to maxima. For the other points, the existence and the value of minima depends on the geometry of the situation:

- Case I: $1 - r_1 - r_2 > 0$, Circles are disjoint.

  Points 1 and 2 are saddle points. Points 3 corresponds to minima of $\delta_{m_1 m_2}$ of value δ_I^* , reached for $(\theta_1, \theta_2) = (0, \pi)$:

$$\delta_I^* = 1 - r_1 - r_2$$

- Case II: $\begin{cases} 1 - r_1 - r_2 \leq 0 \\ 1 + r_1 - r_2 \geq 0 \\ 1 - r_1 + r_2 \geq 0 \end{cases}$, Circles intersect.

  Points 1, 2 and 3 correspond to saddle points of $\delta_{m_1 m_2}$. The system of equations

$$\begin{cases} r_1 sin\theta_1 = r_2 sin\theta_2 \\ 1 - r_1 cos\theta_1 + r_2 cos\theta_2 = 0 \end{cases}$$

  characterizes the triangles whose vertices are the centers of the circles and one of the points of intersection. Minima of value $\delta^*_{II} = 0$ are reached for the angles:

$$\begin{cases} \theta_1^* = \pm Acos(\frac{1+r_1^2-r_2^2}{2r_1})\,(2\pi) \\ \theta_2^* = \pm Acos(-\frac{1-r_1^2+r_2^2}{2r_2})\,(2\pi) \end{cases}$$

- Case III: Circles are included in one another.

  - Case IIIa: $1 + r_1 - r_2 < 0$, $C_1 \subset C_2$

    Points 1 and 3 are saddle points.

    Point 2 corresponds to minima of $\delta_{m_1 m_2}$ of value δ^*_{IIIa} , reached for $(\theta_1, \theta_2) = (\pi, \pi)$:

$$\delta^*_{IIIa} = r_2 - r_1 - 1$$

  - Case IIIb: $1 - r_1 + r_2 < 0$, $C_2 \subset C_1$

    Points 2 and 3 are saddle points.

    Point 1 corresponds to minima of $\delta_{m_1 m_2}$ of value δ^*_{IIIb} , reached for $(\theta_1, \theta_2) = (0, 0)$:

$$\delta^*_{IIIb} = r_1 - r_2 - 1$$

In every case the minima corresponding to a nul differential are absolute minima. If they do not belong to $\mathcal{W}$ the minimum of $\delta_{m_1 m_2}$ is reached on the boundary of $\mathcal{W}$.

$$Boundary(\mathcal{W}) = \cup_{i \in \{a,b,c,d\}}\, \mathcal{B}_i$$

- $\mathcal{B}_a = \{(\theta_1, \theta_2) \in R^2,\ \theta_1 = \Theta_1\ ,\ \theta_2 \in [\Theta_2\ \Theta_2']\ \}$

  $\delta_a = \delta_{m_1 m_2}(\Theta_1, \theta_2)$

$$\frac{d\delta_a^2}{d\theta_2} = 0 \iff -sin\theta_2 + r_1 sin(\theta_2 - \Theta_1) = 0$$
$$\iff t_2^2(r_1 sin\Theta_1) - 2t_2(1 - r_1 cos\Theta_1) - r_1 sin\Theta_1 = 0$$

where $t_2 = \tan\frac{\theta_2}{2}$, $t_2 \in [T_2\ T_2']$. The change of variable is not defined for $\theta_2 = k\pi$, $t_2 = \infty$, which are solutions when $sin\Theta_1 = 0$. In this case the change of variable $t_2' = \tan(\frac{\pi-\theta_2}{2})$ is considered, $t_2' = \frac{1}{t_2}$, which leads to a similar equation.

- If $sin\Theta_1 = 0$ the equations in t_2 and t_2' are of the first degree.

$$\frac{d\delta_a^2}{d\theta_2} = 0 \iff t_2^{(')}(1 - r_1 cos\Theta_1) = 0$$

  * If $1 - r_1 cos\Theta_1 = 0$ (which is possible only for $\Theta_1 = 0\ (2\pi)$) the derivative is nul for all value of θ_2. δ_a is constant.
  * If $1 - r_1 cos\Theta_1 \neq 0$ then $t_2 = t_2' = 0$ for $\theta_2 = 0\ (2\pi)$ or $\theta_2 = \pi\ (2\pi)$:

$$\frac{d^2\delta_a^2}{d\theta_2^2}(k\pi) = -2r_1(1 - r_1 cos\Theta_1)(-1)^k$$

    · If $1 - r_1 cos\Theta_1 < 0$ (which is possible only for $\Theta_1 = 0\ (2\pi)$)
    $\theta_2 = 0\ (2\pi)$ correspond to minima
    $\theta_2 = \pi\ (2\pi)$ correspond to maxima.
    Let $\delta_a^* = \delta(0,0) = \mid 1 - r_1 + r_2 \mid$
    If $\exists k \in Z,\ 2k\pi \in [\Theta_2\ \Theta_2']$, then $min_{\mathcal{B}_a}\delta_a = \delta_a^*$
    else $min_{\mathcal{B}_a}\delta_a = inf(\delta_a(\Theta_2), \delta_a(\Theta_2'))$

    · If $1 - r_1 cos\Theta_1 > 0$
    $\theta_2 = 0\ (2\pi)$ correspond to maxima
    $\theta_2 = \pi\ (2\pi)$ correspond to minima.
    . If $\Theta_1 = 0\ (2\pi)$ then let $\delta_a^* = \delta(0,\pi) = \mid 1 - r_1 - r_2 \mid$
    If $\exists k \in Z,\ (2k+1)\pi \in [\Theta_1\ \Theta_1']$, then $min_{\mathcal{B}_a}\delta_a = \delta_a^*$
    else $min_{\mathcal{B}_a}\delta_a = inf(\delta_a(\Theta_1), \delta(\Theta_1'))$
    . If $\Theta_1 = \pi\ (2\pi)$ then let $\delta_a^* = \delta(\pi,\pi) = \mid r_2 - r_1 - 1 \mid$
    If $\exists k \in Z,\ (2k+1)\pi \in [\Theta_1\ \Theta_1']$, then $min_{\mathcal{B}_a}\delta_a = \delta_a^*$
    else $min_{\mathcal{B}_a}\delta_a = inf(\delta_a(\Theta_1), \delta_a(\Theta_1'))$

- If $sin\Theta_1 \neq 0$ the equation in t_2 is of the second degree.
  $\Delta' = 2(1 - r_1 cos\Theta_1)$

  * If $1 - r_1 cos\Theta_1 \leq 0$ then $\Delta' \leq 0$ and δ_a is monotonous.
    · If $\frac{d\delta_a}{d\theta_2}(\Theta_2) > 0$ then $min_{\mathcal{B}_a}\delta_a = \delta_a(\Theta_2)$
    · If $\frac{d\delta_a}{d\theta_2}(\Theta_2) \leq 0$ then $min_{\mathcal{B}_a}\delta_a = \delta_a(\Theta_2')$

If $1 - r_1 cos\Theta_1 = 0$ then the double root is equal to zero and $\frac{d^2\delta_a}{d\theta_2^2}(0) = 0$. This corresponds to an inflexion point with horizontal tangent.

* If $1 - r_1 cos\Theta_1 > 0$ let $t_2^{\pm} = \frac{1 - r_1 cos\Theta_1 \pm \sqrt{2(1 - r_1 cos\Theta_1)}}{r_1 sin\Theta_1}$

$$t_2^* = t_2^+ \ , \ \delta_a^* = \delta(t_2^+)$$

Depending on the position of t_2^+ several cases are considered:

$$\begin{cases} & t_2^* < T_2 & min_{\mathcal{B}_a}\delta_a = \delta_a(T_2) \\ T_2 \leq & t_2^* \leq T_2' & min_{\mathcal{B}_a}\delta_a = \delta_a^* \\ T_2' < & t_2^* & min_{\mathcal{B}_a}\delta_a = inf(\delta_a(T_2), \delta_a(T_2')) \end{cases}$$

- $\mathcal{B}_b = \{(\theta_1, \theta_2) \in R^2, \ \theta_1 = \Theta_1' \ , \ \theta_2 \in [\Theta_2 \ \Theta_2']\ \}$
  The result is similar to the one obtained for $\mathcal{B}_a$ changing Θ_1 to Θ_1'.

- Since the arcs of circle S_1, S_2 play identical roles, the following cases may be inferred from the previous ones by exchanging the indices and by adding an offset of π to the values of the angles.

  Define as previously:

  $\mathcal{B}_c = \{(\theta_1, \theta_2) \in R^2, \ \theta_1 \in [\Theta_1 \ \Theta_1'] \ , \ \theta_2 = \Theta_2\}$

  $\mathcal{B}_d = \{(\theta_1, \theta_2) \in R^2, \ \theta_1 \in [\Theta_1 \ \Theta_1'] \ , \ \theta_2 = \Theta_2'\}$

$$min_{\ Boundary(\mathcal{W})}\ \delta_{m_1 m_2} = inf_{\ i \in \{a,b,c,d\}}\ (min_{\ \mathcal{B}_i}\ \delta_i)$$

The geometric detection can be implemented in the form of simple mathematical formulas which require only little computation. While geometric filtering was introduced as part of a general detection strategy for efficiency purposes, this geometric detection is rather a back-up method to compensate the untractability of the algebra and to reduce the use of numerical methods.

6.6 Time-Simulation

An analytic approach to conflict detection, involving equations of geometry and kinematics, was given priority over numerical methods whenever possible. After a couple of conditional branchings to select the procedure to apply according to the type of trajectory segment interaction, or to treat separately cases when specific values of the parameters make the computation easier, the expression of the minimum separation during a given time interval is readily available in

a compact form. When the analytic approach becomes too complex, a time-simulation is used.

The State Space representation of flight paths contains all the information which is necessary to infer the precise position and the conditions of flight of an aircraft at any time. A trajectory segment is composed of two states and an operator. The operator gives the law of motion over a time interval. The states correspond to the extreme conditions which constrain the differential equations of motion. For each available operator, the aircraft position is expressed analytically and implemented as a function of the parameters of the transformation, the extreme states, and time. Using the partial application capability of the programming language *Id*, the function is applied to all its arguments but time. A compiler optimization should allow partial evaluation of the closure before arity is satisfied. The time interval of time and altitude overlap is then scanned with a given time increment to obtain the position of the aircraft at various points along the trajectory segment. Distance is computed for the pair of aircraft and checked against the minimum horizontal separation. The size of the time increment must be chosen such that no conflict may occur between two consecutive points of the simulation. If a penetration of the protection volumes by $\frac{1}{20}$ of the three nautical miles horizontal separation is tolerated, and if both aircraft fly at the maximum speed authorized in a terminal area (250 knots), the time increment should not exceed 1.1 second to be sure to detect all conflict:

$$\Delta t_{max} = \frac{\frac{1}{20} sep_H}{2\, v_{max}} = 1.08\ second$$

A time increment of one second is uniformly used though a longer increment would be acceptable for slower flights or when the directions of flight are such that the relative speed is less than the scalar sum of the speeds. For a 180 degree turn with an angular speed of three degrees per second the maneuver lasts one minute. The distance between the aircraft is computed for sixty different values of the time in parallel and in constant time if resources are available.

6.7 Parallelism of the Detection

A significant amount of parallelism is exploited at all the levels of the conflict detection process with various granularities. The hierarchical representation of the flight paths is an opportunity for "data parallelism" while the number crunching involved in the evaluation of mathematical expressions is a source of fine-grained "control parallelism".

Conflict detection can be performed concurrently on different branches of the tree representing a flight path. The detection operates recursively with an increasingly fine granularity. After some logic determines which subparts of a trajectory

which are potentially conflicting with another trajectory are to be tested for conflict, the threads of computation dealing with the different subparts do not interact and can be handled separately. A path is conflict-free if and only if all its subparts are conflict-free. We can not know whether a path is conflict-free before all its subparts have been tested and proved to be conflict-free. On the other hand as soon as any of the subparts involves a conflict we can conclude that the path does involve a conflict and should be rejected or modified. Consequently some time of computation would be wasted if we had to wait until all the answers have flown back up to the root of the tree and all the computation has terminated before reaching a conclusion especially in the cases of conflict. For this purpose some parallel, non-strict logic is used. A binary non-strict AND gate delivers a "false" output as soon as a "false" input has been received. The output is set to "true" when both input are set to "true". The implementation of such a gate uses the same structure as the synchronization mechanisms of the constraint devices which are described in chapter 5. A multiple-input, non-strict AND gate is implemented as a tree of binary gates enhanced by a short-circuiting to handle a "false" input. The result of the test is available as soon as the first conflict, if it exists, has been detected. The state-of-the-art in parallel programming however is such that the threads of computation which go on feeding the input of the logic gate uselessly can not be easily aborted and consume some computational resources. That may affect the global efficiency if these resources happen to be a limiting element for the concurrency of the execution. For the purpose of computational speed, this implementation hides some of the information: only the binary answer is used whereas an analytic detection would easily give the value of the minimum separation, the time of closest approach, and some characteristics of the geometry of the approach. This information could be exploited to differentiate between the conflict-free paths. A more general framework is presented in appendix A. The experience gained by generating and testing the previous paths could be used to direct the generation of new paths. Relying on the speed of both modules, no feedback mechanism has been

The modules of flight path generation and of conflict detection have been implemented in the programming language *Id*. The conflict-free feasible flight paths which are generated and tested in an *Id* environment can be translated into Lisp data structures by using an interface between *Id* and *CommonLisp*. They are then displayed on the screen and simulated in an interactive manner.

6.8 Conclusion

The role of the Conflict Detection module consists of filtering the stream of feasible flight paths which is delivered by the Generation module so as to provide the controller with conflict-free plans. Both modules are designed with a goal of computational efficiency. Conflict detection operates on the flight paths in direct

correspondence with their hierarchical representation to take advantage of increasingly detailed degrees of abstraction. The conflict detection strategy relies on an assumption of conflict sparsity between two trajectories and uses both the notions of kinematic and geometric separation. Several filtering processes are applied successively so as to eliminate the portions of the paths which are conflict-free and focus on the potentially conflicting ones.

"Qualitative rules" embedding high-level properties of the patterns are first used to rule out entire parts of the paths. These rules are by essence pattern-dependent. Since patterns are supposed to evolve, relying too heavily on such rules would make the system inflexible.

Then a "geometric filtering" operates at intermediate levels of abstraction on entire flight phases. Path legs and turns are surrounded by a control volume of extremely simple definition: hypercubes in the spatio-temporal space. By determining the distance between the control volumes this step of filtering ensures geometric separation between large portions of the trajectories. It is computationally inexpensive.

"Kinematic detection" is then applied for a complete detection on the parts which are still potentially conflicting. Analytic expressions of the trajectories are used to determine the minimum separation between trajectory segments. The overlapping time intervals of two sequences of segments are matched. Horizontal separation is only checked on the common portion of the time intervals during which the segments overlap in time and vertical separation is violated. While simple algebraic expressions of the minimum separation are available for the most frequent situations of segment interaction, algebra becomes untractable when arbitrary turn maneuvers are involved. In those limited cases only, two alternative methods are applied: a geometric detection based on algebra, or a time-simulation computing aircraft positions in parallel. Structural and fine-grained parallelism can be exploited extensively at all the steps of the conflict detection.

This method of conflict detection quickly reveals any violation of the separation criteria which may exist between two flight path plans. More might be expected from a Flight Path Generator. In the present situation, paths are either conflicting or conflict-free but there is no nuance in the degree to which two aircraft or trajectories may interact. Such nuances would be useful criteria to differentiate between generated plans which all "look" equally feasible and conflict-free. The implementation of this notion is limited by the lack of expressiveness of the present separation criteria. A model of aircraft interaction is introduced in appendix A as a framework for new definitions of Separation Standards.

Chapter 7

Structure of the Flight Path Generator

This brief chapter summarizes the structure of the Flight Path Generator as it has been implemented, and locates the functional modules in a general schema. Structural extensions involving some feedback and practical suggestions for the integration of the system in controllers' working environment are also presented.

7.1 Structure for a Generate-and-Test Heuristics

The basic components of the generate-and-test heuristics are a Generator and a Test module. The Generator produces a stream of feasible flight path plans which satisfy a given set of operational constraints. The paths fit in the global organization of the terminal area and can be easily flown by the aircraft for which they are generated. The Generator delineates the set of all plausible paths that may be tried as candidate solutions. The Test module filters the stream of feasible paths. It rejects the ones which involve a violation of separation criteria and and accepts the ones which could be proven to be conflict-free.

The basic structure which appears at in figure 7.1.a can be extended to incorporate a Preplanner and a Selector as this is presented in figure 7.1.b. The role of the Preplanner consists of directing the generation process using some domain-dependent knowledge in order to improve the efficiency of the chain. The Selector extracts some preferred paths out of the stream of conflict-free feasible flight path plans. Considering the parallel nature of the implementation of the flight path generation process and of the conflict detection, as presented in chapters 5 and 6, figure 7.1.d more fairly reflects the computational structure of the Flight Path Generator. A functional schema of the Flight Path Generator will be presented in figure 7.2.

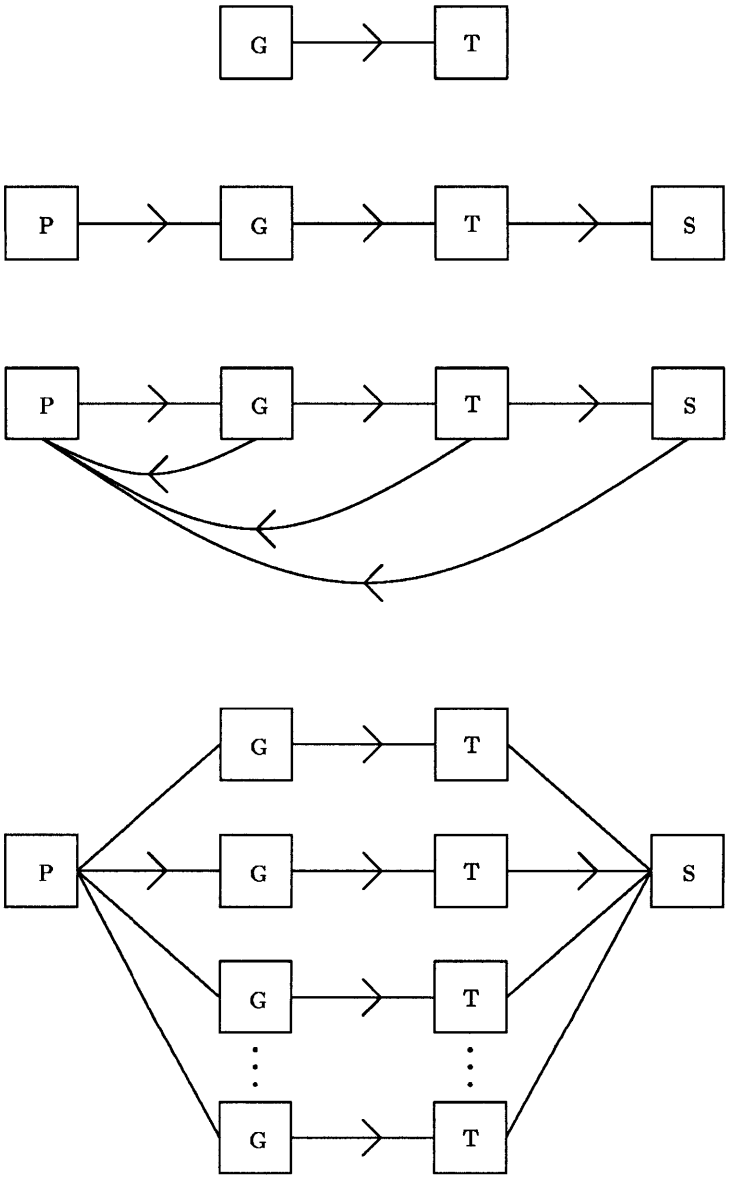

Figure 7.1: Structures for the Generate-and-Test Heuristics

7.1.1 Preplanning and Final Selection

The Flight Path Generator does not generate a unique "best" plan according to some internal criteria but a sample of paths, and gives the opportunity to the user to select among them or to ask for more. In the sample of generated paths, some may be preferable for some strategic reasons or from the point of view of the controller's personal appreciation. The design of the user interface plays a crucial role in the capability of interaction between the controller and the Flight Path Generator. This is a direct consequence of the real-time nature of the interaction with essentially visual information. As candidate paths could be discretely superimposed on the control screen showing digitized displays of radar data, the controller could use a pointing device to select a path. This particular flight path would be highlighted and the controller would then have to press a key to confirm his choice before the path be uplinked.

The purpose of the Preplanner is to restrict the number of attempts in order to minimize the total amount of computation by influencing the generation process towards the production of paths which are feasible, conflict-free, and likely to be selected. Preplanning requires some knowledge about the properties of the flight paths which can be generated. Simple rules, such as trying non-extreme values of the degrees of freedom in priority, increase the probability that a flight path may be built from a given pattern. For example, the "intermediate altitude" could be chosen so as to balance the descent on both the arrival and the downwind leg. An excessively small value would concentrate all the descent phase on the arrival leg. This may require a substantial horizontal distance and prevent the Generator from planning a turn to the desire downwind track because the descent maneuver has not been completed. For a fast aircraft, a long arrival leg and a long base leg may not be possible because of basic geometric limitations. A larger downwind offset may have to be chosen for a fast aircraft than for a slower one.

The following discussion about the integration of some feedback illustrates the interaction of the Preplanner with the rest of the chain of path generation.

7.1.2 Feedback

The general notion of feedback can be used to speed up the convergence of the flight path generation process by taking advantage of some experience gained from previous outputs or attempts. Figure 7.1.c presents a possible structure with feedback for the Flight Path Generator. The following suggestions have not been implemented.

Some feedback from the Generator to the Preplanner may help improving the feasibility of the paths. For instance, assume that the construction of a path from a pattern involving a trombone fails because the length of the "base leg uniform move" segment happens to be negative. The schema in the upper right corner of figure 5.7 corresponds to such a pathological case. That means that

the distance from the runway centerline to the downwind leg does not allow the construction of three turns, an intercept leg involving specified maneuvers, and a minimum length of the base leg between two turns. This minimum distance is expressed as a time between two maneuvers. For fast aircraft it involves a geometric distance which may not be available for small values of the downwind lateral offset. Consequently the construction of a flight path with those values of the degrees of freedom fails. Modifying the direction of the arrival leg, if the pattern is of the type "arrival-trombone", has no chance of success if the value of the downwind offset is maintained. The role of some feedback in this particular example is to indicate to the Preplanner that the downwind offset should be increased. Various values for the direction of the arrival leg may then be tried.

Some feedback from the conflict-detection test module may help building flight paths which avoid obstacles. Assume that two flight path plans which are instances of the pattern "arrival-trombone" are conflicting along their respective arrival leg. Trying to resolve the conflict by modifying the downwind offset of one of the paths would lead to the same conflict. On the other hand, modifying the "intermediate altitude" at which the arrival leg and the downwind leg intersect is likely to separate the aircraft vertically. Horizontal separation may be ensured all along the arrival legs by increasing the angle between the legs. The role of some feedback would consist of indicating to the Preplanner that the directions of the arrival legs should be chosen such as to increase the angle between the legs. Such a conflict might actually be avoided as early as in the preplanning stage. A general strategy would consist of spreading the traffic apart in the three dimensions by taking advantage as much as possible of the available air-space. Practical criteria could be not to trail two aircraft along the same downwind leg if other legs with different downwind offset or altitude are empty, or to evenly dispatch the aircraft arriving from a holding point into the fan composed by the arrival legs with different directions.

7.1.3 Nature of the Constraints

Constraints may be separated into three types depending on the way they are exploited by the generation process.

"Basic constraints" are embedded in the structure of the Generator. They are undisputable rules of algebra or kinematics which are represented at low levels of abstraction and are used to define more expressive compounded constraints.

"Customization constraints" need to be easily interchangeable to adapt the system to a specific environment. The system is adaptable to the specificity of an arbitrary terminal area configuration and can evolve as aircraft performance change with technology. A few flight path patterns like the ones described in chapter 4 should allow controllers to respond to most situations. More efforts should be spent on analyzing the properties of such simple patterns in terms of

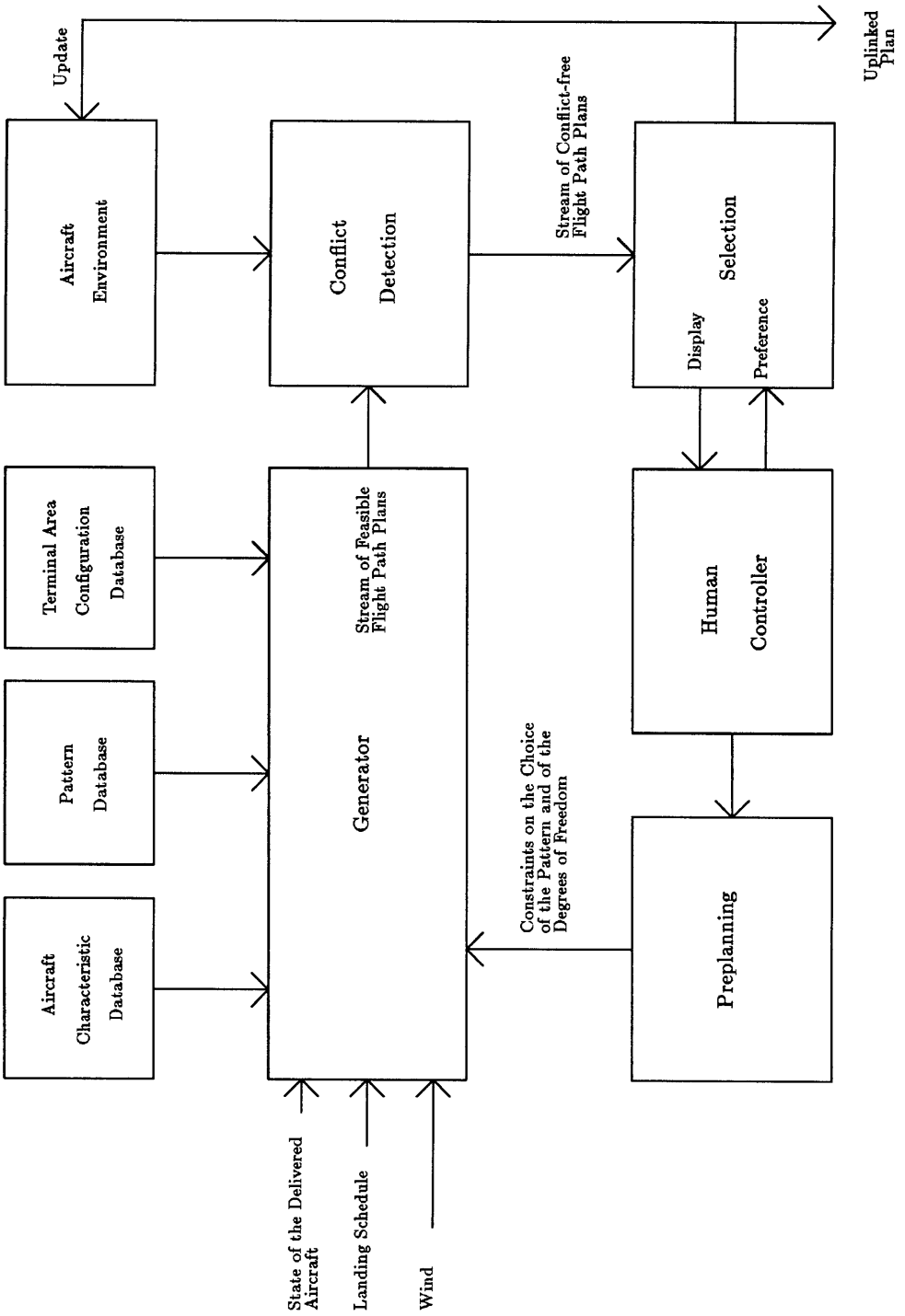

Figure 7.2: General Schema

flexibility, choice of values for the degrees of freedom or pilot workload, than on designing many new patterns. Aircraft are classified in categories with coherent properties according to their aerodynamic characteristics and flight requirements. The available categories are propeller light single, propeller light twin, propeller medium twin, propeller heavy twin and four, turbo-propeller light, turbo-propeller medium, turbo-propeller heavy, jet executive, jet commercial medium, jet commercial standard, jet commercial heavy, supersonic civil. Aircraft models are mapped to a category with the possibility of overwriting some characteristics of the category. A taxonomy may express more sophisticated class inheritance properties. These constraints are treated as data to the flight path generation process. They are kept in databases which are separated from the core mechanisms of construction so that they can be easily updated.

"Dynamic constraints" are those which may be imposed as input to the program. To endeavor to meet pilot requirements, preferred values of the maneuver characteristics could be downlinked and input automatically to the system. In any case, the aircraft characteristics database provides default values. Controllers should also be able to interact with the Flight Path Generator so as to intervene in the decision-making process. A user interface should allow the controller to choose and impose a type of flight path pattern and to fix some degrees of freedom. A menu could appear on the side of the screen when the Flight Path Generator allows or requires controller intervention. The controller could click on the name of a pattern using a mouse device. He would then use a new menu which gives him the opportunity to impose some values of the degrees of freedom by choosing a specific value among the ones which are possible in a given terminal area configuration, or by selecting a subset of these values that he would like to be tried. The usability of such a selection depends a lot on the design of the patterns. A small set of patterns with known geometric and kinematic properties and an evocative name should be available. The degrees of freedom should involve intuitive deformations of the pattern and not be a parameter introduced for mathematical purposes during the design of the pattern.

Chapter 8

Conclusion

A Flight Path Generator is a module of an automated Air Traffic Control system which coordinates the motion of aircraft in the terminal area by a proper planning of flight paths. The problem consists of finding a set of flight path plans which are feasible in the sense that they can be easily flown, conflict-free according to separation criteria, and which fit in the global organization of the terminal area air-space. Creating trajectory plans with a strategic purpose has long been an ill-defined problem which is amenable neither to a mathematical formulation nor to an empirical approach. This work proposes solutions to the following two fundamental issues: structuring the problem by designing a model and choosing a representational framework, and providing a methodology and algorithms for an efficient flight path generation.

The approach which was chosen diverges from all previous attempts of optimal control and focuses on the aspects of air-space organization, on the expressivity of a symbolic modeling environment, and on the speed of computation. The result is the combination of an expressive and flexible knowledge representation framework for aircraft motion in space and flight path plans embedding a structured air-space organization, a fast heuristics based on a constraint propagation paradigm which is supported by parallel computation on dataflow architecture computers, and robust conflict-detection algorithms.

8.1 Knowledge Representation

A well-defined and expressive semantics has been introduced to describe aircraft trajectories. It is used as a framework to design flexible flight path patterns.

First, aircraft motion in space is modeled in the representational framework of a State Space, as a succession of states linked by operators. Operators are declarative statements which represent aircraft maneuvers and can be compared to the "vector commands" given today by *ATC* controllers. The basic building blocks of the trajectories are simple geometric figures and elementary kinematic

actions which can be manipulated symbolically and allow spatio-temporal common sense reasoning.

Patterns are the key-stones of the representation of flight paths. They correspond to the highest degree of abstraction of a hierarchical description of the trajectories. Patterns are defined so as to incorporate pilots' and controllers' experience and to reflect the spatial organization of the terminal area. Each pattern has multiple degrees of freedom which can be adjusted to adapt the path to a particular aircraft type, to meet a landing schedule or other operational constraints, and to satisfy strategic requirements of traffic management. Plan conformance is facilitated by the insertion of time buffers between the maneuvers so as to absorb execution inaccuracies before they accumulate. The relationships between the degrees of freedom which are inherent to a pattern can be expressed as the set of constraints of a linear program. Some traditional mathematical tools are consequently available to study the flexibility of the pattern. They can be applied so as to optimize the deformation of the pattern to accommodate operational deviations, especially delays in the landing schedule.

Several alternative patterns are suggested. They are based on the use of a traditional trombone but also allow short-cutting the downwind leg by turning directly to the base leg, crossing the runway centerline with an overhead maneuver and approaching from the opposite side, or even accommodating a missed-approach by reinserting the aircraft in the landing flow with the possibility of a holding period. These patterns are complementary in the sense that they allow spreading and balancing the traffic in the air-space for strategic purposes. Patterns can be tuned to the specificity of each terminal area and are aimed to be easily interchangeable to allow the system to evolve.

8.2 Flight Path Generation by Parallel Constraint Propagation

The process generating flight path plans is modeled as a Generate-and-Test heuristic. It relies on the fast generation of a stream of feasible flight paths which is filtered to eliminate those which involve any violation of separation criteria. Multiple feasible conflict-free flight path plans can then be generated and presented to the controller in an interactive manner.

Patterns are predefined and adjustable forms to build flight paths through the terminal area. Their use preconstrains the flight path generation process and makes it possible to deal with the infinity of envisionable trajectories. It nevertheless leaves enough freedom for a flexible planning since numerous diversified flight paths with various properties can be generated.

The construction process of a flight path can be viewed as an accumulation of constraints which incrementally specifies the set of feasible trajectories. The

constraint propagation model of computation provides mechanisms to efficiently build flight paths by propagating values in a network of constraints. Abstraction and modularity enhance the intuitivity of the model and help avoiding deadlocks which might result from the locality of the inference process.

Dataflow architecture computers offer a very good match between the programming paradigm and the hardware support. A constraint network is implemented as the superposition of dataflow graphs which are linked by a distributed synchronization mechanism. The implementation exploits the high degree of parallelism and asynchrony which are inherent to the model at the price of introducing some specific-purpose non-determinacy in the clean functional semantics of the programming language *Id*.

8.3 Conflict Detection

The Conflict Detection module filters the stream of feasible flight path plans which are delivered by the Generation module. It relies on the hierarchical representation of the flight paths and on generic procedures to detect all violation of separation criteria between planned trajectories, with the purposes of completeness and computational efficiency.

The detection strategy consists of a cascade of filterings which incrementally focus on the portions of the paths which are potential sources of conflict at a given level of abstraction. Symbolic reasoning, algebraic resolution, and numerical computation are combined recursively with a progression from symbolic to numerical detection. The embedding of specific knowledge about properties of the patterns for a qualitative filtering has been limited to favor an evolving and portable implementation. Analytic expressions of the flight paths are used to determine the minimum separation between flight path segments by considering alternatively the laws of motion over a time interval, or only the geometric extent of each portion of trajectory. A geometric filtering first eliminates large portions of the trajectories. Then a kinematic filtering, which is complemented by a time-simulation when equations are untractable, exhaustively checks the remaining parts.

Several degrees of parallelism are exploited by the conflict detection. For each aircraft entering the terminal area air-space, several flight path plans can be generated in parallel and tested for conflict against the previously existing ones. Besides the tree structure involved in the description of the paths favors concurrency at various levels of abstraction. At the lowest level, the underlying number crunching takes advantage of fine-grained parallelism.

8.4 Extensions

Some practical improvement of the present implementation have been suggested. A Preplanner orders the possible values of the degrees of freedom which are tried according to some knowledge about the configuration of the terminal area. Selection among the sample of all conflict-free and feasible flight path plans which have been generated is essentially performed manually. An interactive display of the paths with attractive graphic capabilities would contribute to the integration of a Flight Path Generator in controllers' routine. The flexibility with which human controllers would be enabled to influence the generation process and select a path might even be viewed by some users as more important than the quality of the paths itself, to the extent that the paths are at least perfectly conflict-free. The internal organization of the Flight Path Generator could be adapted for an enhanced efficiency by structuring the feedback between the modules so as to direct the search depending on the adequacy of previous tries with the goal.

The following appendices present an innovative model of aircraft interaction and a possible extension to the function of Flight Path Generator. A model of aircraft interaction provides a mathematical foundation for a practical tool to estimate and compare the degrees of safety of distributions of aircraft in space. It could be used as a framework to rethink the notion of Separation Standards. It is also suggested that the role of a Flight Path Generator may be extended to plan time windows for landing which could be used by the Runway Scheduler.

The design of a Flight Path Generator is a step forward in the automatization process of Air Traffic Control operations in terminal area. It contributes to an increased capacity of the airport system and to an improved safety.

The implementation of a prototype using dataflow architecture computers is a sign that parallel computing has reached a stage of maturity which allows developing real engineering applications. Parallel computing may in turn alter the engineering way of thinking.

Appendix A

A Model of Aircraft Interaction for Separation Standards

The notion of Separation Standards and a retrospective view on their foundations are presented through an overview of current Air Traffic Control procedures. Improvements in Separation Standards definition for an increased efficiency and flexibility are suggested. A model of interaction between aircraft is introduced which takes into account dynamic flight characteristics. This model questions the traditional approach to Separation Standards and should allow significant increase in the operational capacity of the Air Traffic Control system because of its ability to describe and quantify the degree of interaction of a distribution of aircraft.

A.1 Separation Standards:"State of the Art"

Separating aircraft is the essence of Air Traffic Control. A set of rules, known as Separation Standards, has been devised to ensure the safe separation of aircraft. These rules are agreed upon on an international basis.

The definition of Separation Standards followed the introduction of the primary radar for Air Traffic Control in the late nineteen fifties. Primary radar allowed controllers to see aircraft positions in real time albeit without altitude or identity information which still had to be obtained by radio communication.

Separations Standards were developed with a primary concern of safety and applicability by human controllers according to the technological level of that time. In addition to the efficiency and reliability of guidance, navigation, surveillance and communication systems, other more trivial limitations had to be taken into account. For instance the size of a dot on early radar displays was a limitation to discriminate between close aircraft.

A.1.1 Separation Criteria

Horizontal and vertical separation criteria are established.

Horizontal Separation

There are three types of horizontal separation:

- Longitudinal separation:

  Aircraft following the same path must be flying a given number of minutes apart. This time depends on their relative speed and possible altitude modification maneuvers.

- Lateral separation:

  Lateral separation is established by route structure. Aircraft are required to fly on specified tracks which are separated by a given angle when converging to or diverging from a navigational aid, with minimum distance criteria from the navigational aid.

- Radar separation:

  Unlike longitudinal or lateral separation, radar separation is based on the availability of surveillance equipment to provide the controller with the means of confirming pilots' reports.

  In the case of approach control in the vicinity of the radar antenna, a three-nautical mile basic radar separation is established prior to wake-vortex considerations. Beyond forty miles from the radar a four-mile horizontal separation is required. Additional wake-vortex spacing is required when an aircraft follows a larger aircraft. Aircraft are classified into three categories according to their Maximum Gross Take-Off Weight ($MGTOW$). The additional wake-vortex separation may go up to three nautical miles when a "small" aircraft ($MGTOW < 12500$ lbs) follows a "heavy" one ($MGTOW > 300000$ lbs). Separation is usually expressed as time for take-offs.

Vertical Separation

When horizontal separation criteria are violated, vertical separation rules must be applied. Below flight level 290 (approximately 29000 feet) current altitude separation is one thousand feet if the pilot operates with Instrument Flight Rules (IFR), and five hundred feet with Visual Flight Rules (VFR).

A.1.2 Applicability

The necessity for the controllers to detect potential air-space violations visually and to apply separation criterions with no help from automatic decision-aid tools imposed simple criteria. The definition of Separation Standards relies on an implicit model of interaction between aircraft which allows controllers to easily and almost naturally build up a mind picture of the air-space situation. Horizontal and vertical motions and separations are decoupled. When aircraft are not trailed along a one-dimensional path an isotropic separation is required. The dynamics of the situation is usually ignored. For instance the basic three-nautical mile and one thousand-foot separation simply consists of reserving the air-space contained in a slice of a vertical circular cylinder centered on the aircraft.

A.1.3 A Trade-Off

The interaction between radar performance, atmospheric disturbance, air-crew performance and aircraft dynamics, involves errors that may cause collisions and against which preventive measures must be taken.

To face the uncertainty of the "real world", a buffer area surrounding each aircraft is reserved and protected against another aircraft penetrating it. Consequently large volumes of air-space are currently sterilized to ensure safety.

On the other hand, large buffers result in poor traffic flow capacity. In particular landing approach operations is a critical factor in the operational capacity of the runways at major airports.

The role of Separation Standards is to regulate these buffers by striking a compromise between safety criteria and traffic flow capacity.

Though they have proved their reliability through the years, Separation Standards are questioned by the need to accommodate an increasing flow of traffic.

A.2 Adaptive Separation Standards

A.2.1 Reducing Separation Criteria

Separation criteria are minima. Currently they are based on the worst case scenario of an air-traffic situation to assure a minimum degree of safety at all times. In many specific cases they may be reduced without compromising safety. Whereas reducing the values of the separation criteria would imply a reduction of the level of safety, adapting the separation to a particular situation would homogenize the operational degree of safety.

"Controllability" is the keyword in deciding about a possible reduction. Controllability of aircraft mainly depends on the performance of the airborne navigation and guidance systems, the ground surveillance system and communication.

According to the degree of controllability, separation criteria may be tuned for each pair of aircraft. Separation Standards would become an even more complex set of rules that would take into account additional parameters of the current situation which influence controllability such as the type of aircraft of known navigation capabilities, the weather conditions, the degree of traffic congestion.

A.2.2 A Knowledge-based Approach

Air operations currently depend on controllers' subjective judgement and on their cleverness at estimating separations in space and time. With the increased complexity of separation criteria, controllers should not be expected to manipulate quickly the large set of numerical values corresponding to each situation. Otherwise they would restrict themselves to a subset of easily applicable criteria and would ignore the multiplicity of the possible interactions. For instance, categorizing aircraft according to their maximum gross take-off weight for wake-vortex separation is a step in personalizing standards to increase efficiency, though they have deliberately been kept simple for applicability reasons by human controllers.

The syntax of separation criteria is amenable to a rule definition and implementation. A computer tool in the form of a rule-based knowledge-based system would therefore be very suitable to deal with the structure of knowledge that would result from the definition of more complex Separation Standards. The flexibility of the implementation would allow an incremental definition and hardly quantifiable conditions could be manipulated and reasoned about in a symbolic environment.

The approach involving adaptive Separation Standards is promising for a near future with the help of Expert System technology to help controllers in their task. Progressive insertion of more sophisticated separation criteria can be done with minor disturbance of current practice.

Even subtle combinations of rules to determine the adequate separation would still rely on the underlying model of aircraft interaction which contains inherent limitations. A formal model of the interactions between aircraft is presented below. It provides a basis to rethink the present Separation Standards in order to increase the capacity of the ATC system.

A.3 Model of Interaction between Aircraft

The general-purpose model of aircraft interaction presented herein allows one to describe and quantify a distribution of aircraft in terms of separation between aircraft and air-space occupancy. It provides a mathematical tool to estimate the level of safety of a distribution of aircraft and should contribute to improving the definition of separation criteria.

A.3.1 General Principle

The idea is to associate with each aircraft a field of "influence" characterizing its space occupancy and possible influence on other aircraft. The interactions among the aircraft of a distribution are then entirely characterized by the interaction between their respective field of influence. An estimate of the global degree of interaction internal to the distribution is then possible.

The notion of influence of an aircraft is a means of characterizing the part of the air-space where one does not want other aircraft to be because of the presence of the first aircraft.

A.3.2 "Fancy" Protection Volumes

Traditional approach consists of surrounding each aircraft with a protection volume inside which no other aircraft is allowed to enter. Outside this protection volume anything may happen with respect to the internal aircraft and space organization is left to the controllers' judgement.

The shape of the protection volume has traditionally been for simplicity a circular cylinder centered on the aircraft. Notice for instance that some space is unnecessarily sterilized behind the aircraft. Modifying the shape of the protection volumes should allows one to increase the compactness of aircraft in the air-space while maintaining the level of safety.

Refinement may be obtained by abandoning the two-dimensional isotropy. The shape of the protection volume could be modified to take into account the forward motion of the aircraft. For instance the aircraft could be at the focus of an ellipse whose large axis is parallel to the direction of the motion or the cylinder could be replaced by a flat ellipsoid. The variability of the speed could be taken into account by making the protection volume extensible. Direct proportionality can be obtained by expressing the dimensions of the protection volume as time instead of spatial distance. The volume swept by a traditional cylindrical protection volume during a short projection in time, assuming or not a uniform move, may be an easy way to satisfy this purpose. It would basically be a horizontal cylinder with a rectangular cross section whose length is proportional to the speed of the aircraft.

Refining the shape of the protection volumes is a straight forward extension to the present definitions of separation criteria. Volumes bounded by first or second order surfaces would allow the use of geometric properties to reduce the amount of numerical computation necessary to detect the intersection of two such protection volumes.

A.3.3 Expressiveness of a Multi-State Logic

No matter how sophisticated protection volumes may be, their use relies on the two-state logic of administrative regulations. They represent a threshold of acceptability on the separation between two aircraft (and allow controlling controllers).

In the framework of a binary Logic, the description of a distribution of aircraft in a portion of air-space in terms of separation and occupancy is limited to the number of air-space violations and to the volumetric concentration of aircraft. In the general case of absence of violations, the only remaining criterion to characterize the distribution of aircraft is the number of aircraft. For a given set of aircraft in a given portion of air-space, two distributions would look equivalent even though they may differ in the way aircraft are spread in space (expressed in terms of properties of the local volumetric concentration), and in the distribution of the velocity vectors (directions of flight, speeds).

A multi-state logic would increase the descriptive power of the model. Interactions would be diversified in nature or intensity. The use of the notion of near-miss illustrates the approach consisting of defining several levels of interaction. A near-miss is defined in terms of separation criteria as the penetration of another aircraft in the volume corresponding to a horizontal separation of five hundred feet and a vertical separation of one hundred feet. A near-miss is a stronger interaction than a simple violation of separation criteria in the sense that the former implies the latter with an inclusion relation between the respective protection volumes.

Rather than characterizing a situation that is already known to violate separation standards by introducing some nuances about the degree of violation of the rules, it would be interesting to increase the expressiveness of the model to describe safe situations. Characterizing and quantifying the interactions between aircraft which are close enough to influence one another but not in conflict yet would allow one to anticipate the evolution of a distribution of aircraft to prevent air-space violations. Aircraft influence one another when they get closer because their moves are not independent anymore but depend on their reciprocal reactions to avoid future air-space violations.

A.3.4 Use of Fields of Potential

The notion of spatial influence of an aircraft is represented by a field of potential associated with each aircraft. Aircraft i is the source of the potential $V_i(M)$ at the point M in space. The value of the potential is a numeric indication of the intensity of the influence of the aircraft at a point. The higher the potential at a point, the stronger the interaction with other aircraft which would be at that point, or to state it differently the less one wants another aircraft to be there.

Potentials are a means to express apparently independent notions of interaction between aircraft and to combine them in a generalized framework which allows a quantified representation.

Potentials combine the following notions:

- probability of presence of the aircraft at a position within a time horizon
- disturbing effects caused by the presence of the aircraft such as wake-vortex or supersonic effects

Potential functions must satisfy common-sense properties. The highest value of the potential should occur at the position of the aircraft since the presence of another aircraft there would imply a collision. Far away from the aircraft the potential should be equal to zero since the influence of the aircraft becomes negligible as much in terms of probability of presence in a near future as in terms of disturbing effects. In addition to this boundary condition, convergence conditions will be imposed for mathematical reasons.

An aircraft is represented by its associated potential field. Interactions between aircraft are then expressed in terms of interactions between potential fields. The definition of an energy is introduced to characterize the global degree of interaction of a distribution of aircraft.

A.3.5 Spatial Energy of a Distribution of Aircraft

Consider a spatial distribution of N aircraft in a volume τ. Each aircraft is in a given state which includes its position and relevant dynamic flight characteristics.

A potential V_i , $i \in \{1, ..., N\}$, is associated with each aircraft to describe its interaction with the world.

The fields $\vec{E_i}$ derive from the potentials V_i. Let $\vec{E}$ be the total field resulting from the distribution.

Let W be the energy of the distribution defined as follows, which may be called the spatial energy:

$$W = \frac{1}{2} \int\int\int_{\tau} \vec{E}^2 \, d\tau$$

Linearity assumption:

The total field resulting from the simultaneous presence of all the aircraft is the superposition of the fields corresponding to the presence of an individual aircraft.

$$\vec{E} = \sum_{i=1}^{N} \vec{E_i} \quad and \quad V = \sum_{i=1}^{N} V_i$$

$$W = \frac{1}{2} \int\int\int_{\tau} \vec{E}^2 \, d\tau$$

$$W = \frac{1}{2} \int\int\int_{\tau} (\sum_{i=1}^{N} \vec{E_i})^2 \, d\tau$$

$$W = \frac{1}{2}\int\int\int_\tau (\sum_{i=1}^{N} \vec{E_i}^2 + 2\sum_{i=1}^{N}\sum_{j<i} \vec{E_i}\vec{E_j}\ )\ d\tau$$
$$W = \sum_{i=1}^{N} \frac{1}{2}\int\int\int_\tau \vec{E_i}^2\ d\tau + \sum_{i=1}^{N}\sum_{j<i} \int\int\int_\tau \vec{E_i}\vec{E_j}\ d\tau$$

Let $W_i = \frac{1}{2}\int\int\int_\tau \vec{E_i}^2 d\tau$ be the energy corresponding to an individual aircraft isolated in space.

Let $W_{ij} = \int\int\int_\tau \vec{E_i}\vec{E_j} d\tau$ be the energy of interaction between two aircraft. W_{ij} is equal to W_{ji} and characterizes the spatial interaction for a pair of aircraft.

By analogy with the fundamental laws of physics, a complex interaction between multiple elements is decomposed into elementary pairwise relations.

$$W = \sum_{i=1}^{N} W_i + \sum_{i=1}^{N}\sum_{j<i} W_{ij}$$
$$W = \sum_{individual\ aircraft} W_i + \sum_{pairs\ of\ aircraft} W_{ij}$$

A.3.6 Expression of the Energy as a Function of the Potentials

The expression of the spatial energy will be transformed to get an expression function of the potentials. The fields of potential are the initial data of the problem which allow a concrete interpretation.

$$W_{ij} = \int\int\int_\tau \vec{E_i}\vec{E_j}\ d\tau$$
$$W_{ij} = \int\int\int_\tau \vec{grad}V_i\ \vec{grad}V_j\ d\tau$$

Using the relation: $\vec{grad}G\ \vec{A} = div(G\vec{A}) - G\ div\vec{A}$

$$W_{ij} = \int\int\int_\tau (div(V_i\ \vec{grad}V_j) - V_i\ div(\vec{grad}V_j))\ d\tau$$
$$W_{ij} = \int\int\int_\tau div(V_i\ \vec{grad}V_j)\ d\tau - \int\int\int_\tau V_i \triangle V_j\ d\tau$$

The expression of the energy can be simplified under certain assumptions by using Ostrogradsky theorem: $\int\int\int_\tau div(V_i\ \vec{grad}V_j)\ d\tau = \int\int_\sigma V_i\ \vec{grad}V_j\ d\tau$

Assume that the potential functions decrease sufficiently fast at infinity with a smooth behavior so that the latter integral is equal to zero. This is justified since the influence of an aircraft far away from its position becomes negligible as much in terms of probability of presence as in terms of disturbing effects.

If potentials are taken to be equal to zero outside a region surrounding the aircraft the integral is obviously equal to zero. If potential functions decrease like $\frac{1}{r}$ in the neighborhood of infinity, gradients decrease like $\frac{1}{r^2}$ and the integral over a sphere whose surface increases like r^2 is equal to zero.

The expression of the spatial energy becomes:

$$W_{ij} = -\int\int\int_{\tau} V_i \triangle V_j \, d\tau$$

Note that the expression is also valid in the case $i = j$ with $W_i = \frac{1}{2} W_{ii}$.

Considering the symmetric roles played by the two aircraft in the interaction, the final expression of the spatial energy of interaction becomes:

$$W_{ij} = -\frac{1}{2}\int\int\int_{\tau} (V_i \triangle V_j + V_j \triangle V_i) \, d\tau$$

A.3.7 Minimum Energy Criterion

As will be seen in the following example, the goal to is minimize the energy of interaction or to keep it under a defined threshold to intuitively reduce the risk of collision. Similarly the total energy of spatial interaction characterizes the compactness of the distribution and should be reduced to improve the global degree of safety.

Simply minimizing the total energy would lead to inconsistencies. Such a criterion would make it possible to sacrifice a pair of aircraft to increase the separations between the majority of the others. The energies of interaction describe interactions which are internal to a distribution with a thinner granularity. It should be used to ensure pairwise separations.

A set of trajectories that would minimize the energies of interaction at all time would certainly be very safe in terms of separation but would lead to trajectories that can not be flown, essentially because of their constantly changing curvature. Trajectories must be easily flown and fit in the global organization of the air-space.

The notion of minimum energy of a distribution of aircraft is a descriptive property and should be used as a differentiation criterion to estimate and compare the global safety of distributions.

A.3.8 Simple Analytic Example

An example of definition of potential field and the analytic expression of the corresponding energy of interaction are presented. For simplicity, only two dimensions are considered first.

Aircraft i positioned at the point M_i is the source of the potential V_i, where r_i denotes the distance from M_i and α_i is a constant:

$$V_i = \frac{\alpha_i}{r_i}$$

The potential function satisfies the common-sense properties of high (actually infinite) value close to the aircraft and of convergence to zero far from the aircraft where its influence becomes negligible.

To allow analogies with present separation criteria the isotropy of the field around the aircraft was conserved. Note also that an harmonic function was chosen to simplify the expression of the Laplacian.

The expression of the Laplacian is the following:

$$\triangle V_i = -4\pi\alpha_i\delta_{M_i}$$

where δ_{M_i} denotes the impulsion function centered at M_i. The expression of the spatial energy of interaction is:

$$\begin{aligned} W_{ij} &= -\frac{1}{2}\int\int_\tau (\, V_i \triangle V_j + V_j \triangle V_i \,)\, d\tau \\ W_{ij} &= \frac{1}{2}(4\pi\alpha_i\alpha_j)\int\int_\tau (\, \frac{\delta_{M_j}}{r_i} + \frac{\delta_{M_i}}{r_j}\,)\, d\tau \\ W_{ij} &= 4\pi\alpha_i\alpha_j \frac{1}{r_{ij}} \end{aligned}$$

where r_{ij} denotes the distance between aircraft i and j.

According to present Separation Standards a minimum horizontal separation is required:

$$r_{ij} > r_{min}$$

This rule may be transformed to express the separation requirement as a constraint imposed on the energy of interaction:

$$W_{ij} < 4\pi\alpha_i\alpha_j \frac{1}{r_{min}}$$

A minimum distance is translated into an upper bound on the value of the energy of interaction since the reciprocal influence between the aircraft increases as they get closer.

This example has the advantage of showing that traditional separation criteria can be easily expressed in terms of energy of interaction, which supports the intuition of the generality of the model.

To extend this example to a third dimension it appears as a reasonable simplification to decouple the effects of horizontal and vertical separation as it is

presently done in the definition of Separation Standards. The analytic expression of the potential would take the following general form in polar coordinates:

$$V_i = \mathcal{H}_i(r,\theta)\ \mathcal{V}_i(z)$$

Such a form would accommodate the guidelines to define potential fields that will be suggested below.

A.3.9 Tractability

Potentials can be defined analytically or numerically. Numerical methods may practically allow more flexibility in the definition of the potential fields. The form of the expression of the energy is especially suitable to the application of the usual numerical calculus methods.

In the case of a numerical approach, space is quantized and the value of the potential V_i is given at the vertices of a three dimensional grid. Finite difference methods give the values of $\triangle V_i$ as linear combinations of values of V_i at surrounding vertices. The value of the energies W_{ij} are obtained by summing the values of the integrated function for every elementary volume. More sophisticated methods using the translation of multipol or Taylor expansions may prove to be more efficient. The values of the energy of interaction defined for each pair of aircraft may be kept in a matrix form (the energy matrix is positive definite). The total energy is then obtained by computing the associated norm of a unit vector $W = u^t[W_{ij}]u$.

With no additional computational effort, each aircraft may be associated with a weight which would reflect the degree of attention to be given to the aircraft in the sense that it would allow one to tune the importance of the aircraft in its interactions with other aircraft and would have the same effect as a multiplicative factor applied to the definition of the corresponding field of potential. The energy would then be the norm associated to the energy matrix of the weight vector.

A.3.10 Guidelines for Defining a Potential Field

To combine the effects of phenomena of different nature the potential function associated to an aircraft is defined as a linear combination of the elementary potential functions corresponding to each phenomenon which are weighted and summed. This representation allows one to easily consider a new parameter and to adjust the relative importance of the phenomena according to technological evolutions.

Probability of Presence

The probability of presence of an aircraft at a position in space as it is seen by the ground controller is the result of the errors introduced by the surveillance and the flight guidance systems in the determination of aircraft position.

Current *ATC* system measures altitude with in general a hundred foot precision as reported by aircraft transponders. The altitude coding is designed to identify one hundred foot increments based on a fixed standard sea level pressure and the basic error of the altimeter of a transponder varies from twenty feet at sea level to more than hundred feet at twenty thousand feet.

The combination of the surveillance altitude error and of the altitude keeping error of the flight guidance system, assuming that the aircraft is flying level, may be approximated by a normal distribution which is entirely defined by its first two moments, an esperance equal to zero and a standard deviation of one hundred feet.

The measurement of range and azimuth by surveillance radars degrades with distance from radar site. The loci of error of equal probability in the determination of aircraft horizontal position are ellipses with a radial small axis. A difficulty is that surveillance position error varies with position. An upper bound for an airspace area may be an acceptable simplification.

Equipotentials are the surfaces of equal probability of presence which in first approximation have an elliptical horizontal projection and a gaussian vertical profile.

Projection in Time

Anticipating the move of an aircraft and estimating its probability of presence after a projection in time are attempts to master uncertainty of the future to avoid later conflicts.

An equipotential is defined as the locus of the points in space that can be reached by an aircraft within a given time horizon considering standard maneuvers or extreme flight capabilities and the possible conformance with a plan.

If no plan of the pilot's intentions is known, the future of the aircraft is only limited by its flight capabilities in the framework of *ATC* procedures. In the horizontal plane the locus of the points corresponding to an equal time projection is as follows. In the case of a short term projection it is a curve whose endpoints are on the circles with radius corresponding to the maximum rate of turn and which contains a point on the axis of symmetry corresponding to a straight-line motion. For a longer term projection curves are circles centered on the present projection of the aircraft whose radius increases with time.

If the aircraft has been required to conform with a flight path plan, the accuracy of airborne guidance equipment which may vary in a very wide range is determinant. The probability of blunder by pilots in the use of the navigation and guidance equipments may have to be added to instrument errors at maintaining a direction or a constant speed.

Disturbing Effects

The disturbing effect potential is intended to gather all the negative consequences caused by the presence and the motion of an aircraft on other aircraft.

Air turbulences which are the main disturbing effect depend on the type of aircraft and other more situation dependent factors such as the speed or the proximity to the ground.

The area corresponding to an equal intensity of turbulence, characterized by an overpressure or a degree of irregularities in the flow, may be used to define equipotentials. Because of the complexity of the fluid dynamics involved, precomputed patterns of turbulence could be used for different phases of the flight.

Fixed obstacles which have to be avoided by the flying aircraft can be represented in the framework by the combination of fields of potential indicating their spatial occupancy and characterizing their effects on the neighboring traffic.

A.3.11 A Practical Example

The expressiveness of the framework based on the use of potential fields allows the description of a wide spectrum of interaction between aircraft. A difficulty may be to formulate the problem in terms of appropriate potentials so as to reflect the influence of the various interaction effects. Another concern may be the computational expense incurred in a dynamic situation to provide a value of the energy of interaction as a function of time for flight path planning purposes, when the relative positions of the aircraft, and consequently of the associated potential fields, are constantly changing. Induced potentials may also have to be considered to account for the fact that the effect of an aircraft on the outside world may be induced by the presence of others.

The following example is intended to demonstrate the practically of the method. It shows how a simple potential function which has a tangible physical meaning can be defined and how it can be expressed with a goal of computational efficiency.

Equipotential surfaces are a family of ellipsoids with a common focus F corresponding to the estimated position of the aircraft. The longitudinal dimension of the ellipsoids is proportional to the air-speed of the aircraft. This is a way of representing the dynamics of the situation by account for the probability of presence of the aircraft at a future time. The factor of proportionality may be increased if the aircraft is accelerating. Similarly the surfaces may be curved in the case of a turn. Such sophistications may take advantage of the knowledge of the flight path plan to tune the interaction to the expected behavior of the aircraft.

Let F, F' denote the foci of an ellipsoid, $\vec{S}$ the aircraft air-speed and k a proportionality factor:

$$\vec{FF'} = k\,\vec{S}$$

The points M belonging to the same ellipsoid satisfy:

$$\overline{FM}^2 + \overline{F'M}^2 = d^2$$

where d represents the size factor characterizing each member of the family of surfaces. The potential on the surface is a function of d : $V_M = V(d)$. An analogy with a result of electrostatics is going to help express the potential function. The equipotential surfaces of the electrical field created by a uniformly charged straight-line segment are ellipsoids whose foci are the segment end-points. The expression of the potential on the line going through the foci is the following:

$$V(x) = \lambda \ln(|\frac{x+c}{x-c}|)$$

where λ is the density of charge, $2c = |\overline{FF'}|$, and x denotes the distance to the middle of the segment which is related to d by the relationship: $2(x^2 + c^2) = d^2$. This expression naturally satisfies the convergence condition since $V = \theta(\frac{1}{x})$ in the neighborhood of infinity. The interactions of the field of influence of the aircraft are entirely characterized by the interactions of a charged segment which is the source of an identical potential field.

The analytic expression of the energy of interaction between uniformly charged straight-line segments is not simple. To avoid the computational cost of spatial integrations, the potential field can be approximated as the potential created by a distribution consisting of discrete charges. The potential function can be approximated with any desired precision by concentrating the total charge $(2\lambda c)$ of the segment at a finite number p of points which are equally spaced along the segment. The expression of the energy of interaction for any pair of aircraft $\{1, 2\}$ is the following:

$$W_{12} = 4\pi \frac{2\lambda_1 c_1}{p_1} \frac{2\lambda_2 c_2}{p_2} \sum\sum_{\substack{1 \le i \le p_1 \\ 1 \le j \le p_2}} \frac{1}{r_{ij}}$$

The interaction between the two aircraft can be summerized by a matrix of dimension $p_1 \times p_2$ containing the inverse of the distances for each pair of discrete charges. The static potential for a fixed distribution of aircraft can be computed at low cost by approximating it as the influence of a small number of discrete sources. The dynamic problem can then be treated by repeating the computation of the static potential with a frequency which will be dictated by the needs of conflict detection algorithms.

A.4 Conclusion

The need to accommodate an increasing flow of traffic questions the present Separation Standards which were defined as simple rules applicable by human con-

trollers but in the absence of a general framework to study the validity of the separation criteria.

Considering the increased aircraft controllability, a temptation is to reduce separation criteria. Whereas a global reduction would compromise the warrantied level of safety, adapting the separation to each situation would homogenize the degree of safety of ATC operations. Expert system technology is a promising approach to help controllers manipulate new more complex criteria.

A general model of aircraft interaction has been introduced. It provides a tool to describe and quantify the degree of interaction of a distribution of aircraft. The model has a mathematical foundation and is amenable to numerical methods.

The influence of an aircraft on its environment is represented by a field of potential. Guidelines are presented to define fields of potential which combine influences of various nature such as the notion of probability of presence of the aircraft, a projection in time of aircraft behavior or their disturbing effects on the air-space environment. Approximating potential functions by the use of discrete sources limits the computational expense for a dynamic approach.

An energy of interaction is defined as the integral over the air-space of local interactions between the potential fields. An energy matrix gathers the interactions that are internal to a distribution of aircraft. The total energy of spatial interaction of a distribution is a macroscopic quantity which characterizes the occupancy of the air-space and the compactness of a distribution while taking into account the dynamics of the situation. A minimum energy criterion may be used to estimate and compare the degree of safety of distributions.

Appendix B

Time Window for Landing

B.1 Earliest Possible Landing Time

Given an arriving aircraft which is delivered to the planning system at an initial position with initial kinematic conditions, and a spatial environment, the Earliest Possible Landing Time ($EPLT$) is the earliest time at which the aircraft may reach the runway for landing with respect to the environment. It is the earliest time such that there exists a satisfactory flight path that leads the arriving aircraft to the runway at that time. A satisfactory path should be feasible with respect to ATC procedures and navigation capabilities and conflict-free with respect to the environment. Consequently finding the $EPLT$ requires proving the existence of such a path and implicitly being able to provide it.

The determination of the necessary $EPLT$ is necessary to provide the Runway Scheduler with a lower bound on the possible landing time to avoid inconsistent schedules that are not even materially achievable. The $EPLT$ is a binding constraint on the runway scheduling process when the rate of arrival is small. As the rate increases, the earliest possible landing time decreasingly constrains the schedule. With a high arrival rate large queues build up and queued aircraft are readily available for immediate operation. When the system is saturated by a high arrival rate, which is the most demanding case for optimizing the ATC procedures, the $EPLT$ is not a binding constraint.

B.2 Latest Possible Landing Time

In spite of symmetrical definitions, finding the Latest Possible Landing Time ($LPLT$) does not involve the same problems as for the $EPLT$. Thanks to the use of holding fixes arriving aircraft may be held for an indefinite amount of time in isolation from other traffic. The upper bound on the landing time comes from the aircraft itself and may correspond to the time when it runs out of fuel.

The $LPLT$ is luckily not a binding constraint in practice. However to accommodate such cases of emergency the runway scheduling process should be able to assign priorities.

B.3 Constraints on the $EPLT$

In order to determine the value of the $EPLT$ it would be pleasant to have a means of directly getting a flight path that is known to the shortest possible path for a given situation. Criteria would be used such as the following: the shortest path between two positions may simply be composed of a straight-line linked to an arc of circle of minimal radius at each extremity to compensate for a change in heading. However the $EPLT$ depends on the flight path generation process and the minimal value of a possible landing time must correspond to an operationally feasible path that the Flight Path Generator knows how to generate with respect to all the constraints imposed on the path. For example, if the aircraft is at a high altitude, a longer path may be required to allow time for descent.

The first requirement is the existence of an operationally feasible flight path to the runway. This depends on the choice of the flight path patterns which are made available to the Flight Path Generator for a given aircraft and terminal area air-space organization. Independently of other traffic the aircraft must at least be able to reach the runway at the chosen landing time. There are material limitations on the time of arrival at the runway considering the total length and the shape of the possible instances of the patterns along with a bounded speed. To be actually flown a flight path must also be conflict-free. If the $EPLT$ is underestimated it leaves the door open to runway schedules which can not be met because no satisfying path can be generated. A new schedule would have to be defined and a new set of flight paths generated. If the $EPLT$ is overestimated some perfectly acceptable schedules are ruled out. A new source of suboptimality for the runway schedule is introduced.

In fact, if one expects the $EPLT$ to take into account potential conflicts, the time window which constraints the runway scheduling process may not be a convex interval between the earliest and the latest landing time but the union of several such intervals. For instance it may be possible to take advantage of some empty space to meet an early schedule but this space may happen to be occupied by an other aircraft later on and no other path feasible for a while. That implies a discontinuity of the time window.

The motivation to consider the influence of conflicts on the $EPLT$ is to be able to provide the runway scheduler with a time interval, as long and less constraining as possible, during which it is known that a conflict-free path can be generated for a given aircraft. Considering potential conflicts in an environment with multiple aircraft would introduce arbitrary discontinuities in the process of finding the $EPLT$ and would significantly increase its complexity what may not be justified

by the need for an accurate determination of the $EPLT$.

B.4 Suggestions for Determining the $EPLT$

Suggestions to estimate the $EPLT$ take advantage of properties of the flight path patterns that are independent of any other traffic. Considering that the choice of a pattern determines the geometric shape of the path and the succession of maneuvers, some patterns are more likely to lead to the shortest paths than others. Patterns involving a smaller number of phases, maneuvers or legs should allow faster arrival at the runway. The pattern "arrival-trombone" is more time-demanding than the pattern "arrival-direct-to-base" when they are both applicable because of the additional downwind leg which is skipped in the latter case to turn directly to a base leg. Flying the pattern "overhead-trombone" requires more time than for the former two because of the need to change side with respect to the runway centerline after first converging towards an axis situated close to the runway. Patterns may be classified according to a relative estimate of the time required to fly them, and then tried sequentially if the previous ones have failed. If two patterns are so different that no qualitative comparison is possible it is envisionable to try both in parallel and to take the minimum of the $EPLT$'s with each pattern.

The way the paths are built, the maneuver characteristics are imposed first to satisfy necessary speed or altitude constraints. Assuming the deceleration and the descent rates are fixed as maneuver requirements, the total time allocated for these maneuvers is imposed by the initial speed and altitude. The number of turns and the angles are characteristics of the patterns. Short uniform-move segments are inserted to ensure a minimum separation between the maneuvers and reduce pilot workload. Flight paths are completed by stretching and twisting the remaining path segments for geometry and time adjustments. The total duration of the maneuvers is imposed. At this stage the goal is to minimize the duration of the uniform moves which link the maneuver phases. The choice of values for the degrees of freedom may first rely on simple empirical observations about the patterns which are made available to the Flight Path Generator. For a given pattern, examining the various combinations of values for the degrees of freedom should allow an empirical classification of the possible instances of the pattern with respect to their expected duration without having to entirely generate them. For a more mathematical approach, the total duration of a flight path can be expressed as a function of the degrees of freedom. The problem consists of minimizing a multi-variable function involving a mix of arithmetics and trigonometry, when some variables vary continuously and others are quantized.

The tools which have been designed to build and generate flight path plans should be applicable to determine the $EPLT$. The flight path generation process is constrained at both extremities of the flight path plan by the delivery time and the landing time. These constraints are propagated in both directions along the

plan. For determining the $EPLT$ only the delivery time is imposed. The forward character of the propagation is accentuated. As the flight path is being constructed degrees of freedom are progressively fixed with the goal to reach the runway as soon as possible. If an inconsistency is detected during the propagation process the requirement of earliest possible landing is relaxed and alternative values for the degrees of freedom are chosen with the same policy. The landing time is eventually instantiated with a value which is taken as the $EPLT$.

Bibliography

Flight Transportation, Air Traffic Control

[1] *Airport Capacity Enhancement Plan Report*, Report DOT/FAA-CP-86-1, FAA, U.S. Department of Transportation, 1986

[2] Analytical Mechanics Associates, *OPTIM: Computer Program to Generate a Vertical Profile which Minimizes the Aircraft Direct Operating Costs*, Revision No. 2, NAS1-1549, Inc. , Montain View, CA. October 1980

[3] Analytical Mechanics Associates, *TRAGEN: Computer Program to Stimulate an Aircraft Steered to Follow a Specified Vertical Profile*, Revision No. 2, NAS1-1549, Inc. , Montain View, CA. October 1980

[4] Athans M. and Porter L.W. , *An Approach to Semi-Automated Scheduling and Holding Strategies for Air Traffic Control*, 1971 Joint Automatic Control Conference Preprints, pp. 536-545, 1971

[5] Benoit A. , Swierstra S. , *Available Tools for the Prediction, Control and Economy Assessment of Flight Profiles* EUROCONTROL, Division E1, Doc 822033, 1982

[6] Bowles R. and Mercer G. , *Method for Forecasting Commercial Traffic*, FAA, U.S. Departement of Transportation, February 1987

[7] Cohen D.A. and Odoni A.R. , *A Survey of Approaches to the Airport Slot Allocation Problem*, Flight Transportation Laboratory Report, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, 1985

[8] Day C.F. and White J.M. , *A Slot Allocation Model for High Density Airports*, FAA Office of Aviation Policy and Plans, FAA-APO-80-13, 1980

[9] Dear R.G. , *The Dynamic Scheduling of Aircraft in the Near Terminal Area*, Flight Transportation Laboratory Report, R76-9, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, 1976

[10] Erzberger H. and Lee H.Q. , *Optimal Horizontal Guidance Techniques for Aircraft*, Journal of Aircraft, Vol. 8, No. 2, p. 95, 1971

[11] Erzberger H. and Lee H.O. , *Characteristics of Constrained Optimum Trajectories with Specified Range*, NASA TM-78519, September 1978

[12] Erzberger H. and Lee H.Q. , *Terminal-Area Guidance Algorithms for Automated Air-Traffic Control*, Ames Research, NASA TN D-6773, 1972

[13] *FAA Aviation Forecasts, Fiscal Years 1987-88*, report FAA-APO-87-1, FAA, Department of Transportation, February 1987

[14] Flight Transportation Associates, *Development of Scheduling and Flight Path Generation Algorithm for MATCALS*, Technical Report to Naval Electronics Command, Washinghton D.C. , March 1983

[15] *Future Development of the U.S. Airport Network: Preliminary Report and Recommended Study Plan*, National Research Council (U.S.), Transportation Research Board, 1988

[16] *Instrument Approach Procedures*, National Oceanic and Atmospheric Administration, U.S. Department of Commerce, published every year

[17] Hsin C. , *An Analytical Study of Advanced Terminal Area Traffic Management and Control*, Flight Transportation Laboratory Report, R76-13, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, 1976

[18] Knox C.E. , Cannon D.G. , *Development and Test Results of a Flight Management Algorithm for Fuel-conservative Descents in a Time-based metered Traffic Environment*, NASA Technical Paper 1717, October 1980

[19] Knox C.E. , Vicroy D.D. , Simmon D.A. , *Planning Fuel-Conservative Descent in an Airline Environment Using a Small Programmable Calculator*, NASA, 1985

[20] Lee H.W. and Erzberger H. , *Algorithm for Fixed-range Optimal Trajectories*, NASA Technical Paper 1565, July 1980

[21] Lee H.Q. , *An Advisory System for Predicting and Resolving Airspace Violations based on Four-dimentional Guidance Techniques*, NASA Ames Research Center, Moffett Field California, AIAA Guidance, Navigation and Control Conference, Monterey California, August 1987

[22] Malherbe G. , *Modeling of Wind and Radar for Simulation in Four-Dimension Navigation Environment*, Flight Transportation Laboratory Report, R78-8, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, 1976

[23] Papadimitriou C.H. , Steiglitz K. , *Combinatorial Optimization: Algorithms and Complexity*, Prentice Hall, 1982

[24] Pararas J.D. , *Decision Support for Automated Terminal Area Air Traffic Control*, Flight Transportation Laboratory Report, R82-3, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, 1982

[25] Sadoune M.M. , *An Expert System for Generating Terminal Area Flight Paths for Arriving Aircraft*, Flight Transportation Laboratory Report, R87-8, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, July 1987

[26] Sadoune M.M. , *A Model of Aircraft Interaction for Separation Standards*, Flight Transportation Laboratory Memo., M88-3, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, April 1988

[27] Sadoune M.M. , *Functional Description of ATC Operations in the Terminal Area*, Flight Transportation Laboratory Memo., M88-4, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, June 1988

[28] Sadoune M.M. , *Flight Path Patterns*, Flight Transportation Laboratory Memo., M88-5, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, July 1988

[29] Sarris A.H. and Athans M. , *An Algorithm for Terminal Area Air Traffic Control*, Report ESL-P-466, Electronic Systems Laboratory, Massachusetts Institute of Technology, December 1971

[30] Schmidt D.K. and Swain R.L. , *An Optimal Control Approach to Terminal Area Air Traffic Control*, Journal of Aircraft, Vol. 10, No. 3, pp. 181-188, March 1973

[31] Schultz R.L. and KilpatrickP.S. , *Aircraft Optimum Multiple Flight Path*, JANAIR Rept. 700709, Honeywell Inc. , Minneapolis, Minn. , 1970

[32] Simpson R.W. , *An Analytical Description of Air Traffic Control Systems*, Course notes, Flight Transportation Laboratory, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, 1985

[33] Simpson R.W. , *Expert Systems for the Generation of Terminal Area Arrival Paths for Civil Transport*, Aircraft Trajectories Computation, Prediction and Control, AGARDgraph AG-301, 1988

[34] Sorensen J.A. , Waters M.H. , *Generation of Optimum Vertical Profiles for an Advanced Flight Management System*, NASA Contractor Report 165674, March 1981

[35] Sorensen J.A. , *Design and Analysis of Advanced Flight Planning Concepts*, NASA Contractor Report 4063, March 1987

[36] *Standard Operating Procedures*, Federal Aviation Administration, U.S. Department of Transportation, Order BOS-TWR-7110.35E, 1986

[37] Talley J.R. , *Basic Metering and Spacing for ARTC III*, SRDS Progress Report, DOT-FAA, Washington D.C., 1978

[38] Telson M.L. , *An Approach to Optimal Air Traffic Control During the Landing Phase*, M.S. Thesis, Dept. of Electrical Engineering, Masssachusetts Institute of Technology, 1969

[39] *Terminal Area Forecasts: Fiscal Years 1987-2000*, Report FAA-APO-87-8, FAA, U.S. Department of Transportation, April 1987

[40] Tobias L. , *Automated Aircraft Scheduling Methods in the Near Terminal Area*, Journal of Aircraft, Vol. 9, No. 8, pp.520-524, 1972

[41] Tosic V. , Horonjeff R. , *Effect of Multiple Path Approach Procedures on Runway Landing Capacity*, Institute of Transportation and Traffic Engineering, University of California, Berkeley, California, 1975

[42] Trivizas D.A. , *Parallel Parametric Combinatorial Search - Its Application to runway scheduling*, Flight Transportation Laboratory Report, M87-4, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, 1987

[43] Wilz R.R. Wanock A. , *MATCALS: Expansion of Capacity for Expeditionary Airfields*, Paper at the twentieth Guidance and Control Panel Symposium, AGARD Meeting, Cambridge, Massachusetts, May 1975

[44] Allen J.F. , *An Interval Based Representation of Temporal Knowledge*, Proceedings of IJCAI-7, 1981

[45] Allen J.F. , *Maintaining Knowledge about Temporal Knowledge*, Communication of the ACM 26-11, 1983

[46] Arvind , Gostelow K. and Plouffe W. , *An Asynchronous Programming Language and Computing Machine*, Technical Report TR-114a, Department of Information and Computer Science, University of California, Irvine, CA, December 1978

[47] Arvind and Thomas R.E. , *I-structures: An Efficient Data Type for Parallel Machines*, Technical Report TM-178, Computation Structure Group, Laboratory for Computer Science, Massachusetts Institute of Technology, 1980

[48] Arvind and Gostelow K. , *The U-Interpreter*, Computer, 15 (2), February 1982

[49] Arvind and Culler D.E. , *Dataflow Architectures*, Annual Reviews in Computer Science, pp. 225-253, Annual Review Inc., Palo Alto, CA, February 1986

[50] Arvind, Nikhil R.S. , Pingali K.K. , *I-structures: Data Structures for Parallel Computing*, Proceedings of the Workshop on Graph Reduction, Los Alamos, New Mexico, February 1986

[51] Arvind , Nikhil R.S. , *Executing a Program on the MIT Tagged-Token Dataflow Architecture*, Proceedings of PARLE (Parallel Architecture and Languages, Europe), Eindhoven, The Netherlands, June 1987

[52] Arvind, Dertouzos M.L. , Nikhil R.S. , Papadopoulos G.M. , *Project Dataflow - A Parallel Computing System Based on the Monsoon Architecture and the Id Programming Language*, Memo. 285, Computation Structure Group, Laboratory for Computer Science, Massachusetts Institute of Technology, March 1988

[53] Blelloch G. , *Applications and Algorithms on the Connection Machine*, Report TR87-1, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1987

[54] Dennis J.B. , *First Version of a Data Flow Procedure Language*, in G.Goos and J.Hartmanis editors, Proceedings of the Programming Symposium, Paris 1974, Springer-Verlag, Berlin, 1974, Lecture Notes in Computer Science 19, 1974

[55] Doyle J. , *A Truth Maintenance System*, Artificial Intelligence 12, pp. 231-272, 1979

[56] Feldman Y.A. , Rich C. , *The interaction between Truth Maintenance, Equality and Pattern-Directed Invocation: Issues of Completeness and Efficiency*, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1987

[57] Gurd J.R. , Kirkham C. and Watson I. , *The Manchester Prototype Dataflow Computer*, Communications of the ACM, 28(1):34-52, January 1985

[58] Hillis W.D. , *The Connection Machine*, MIT Press, Cambridge, Massachusetts, 1985

[59] Hiraki K. , Sekiguchi S. , Shimada T. , *System Architecture of a Dataflow Computer*, Technical Report, Computer Systems Division, Electrotechnical Laboratory, 1-1-4 Umescono, Sakuramura, Niiharigum, Ibaraki 305, Japan, 1987

[60] Kowalski R.A. , *Algorithm = Logic + Control*, CAMC 22, 1979

[61] Long W.J. , Russ T.A. , *A Control Structure for Time Dependent Reasoning*, Laboratory for Computer Science, Massachusetts Institute of Technology, Proceedings of IJCAI-83, 1983

[62] Martelli A. , Montanari U. , *An Efficient Unification Algorithm*, ACM Trans. on Programming Languages and Systems, 4, 1982

[63] Nikhil R.S. , *Id Reference Manuel (version 88.0)*, Memo 284, Computation Structure Group, Massachusetts Institute of Technology, Laboratory for Computer Science, March 1988

[64] Paterson M.S. , Wegman M.N. , *Linear Unification*, Journal of Computer and Systems Science 16, 1978

[65] Papadopoulos G.M. , *Implementation of a General-Purpose Dataflow Multiprocessor*, Ph.D. Thesis, Laboratory for Computer Science, Massachusetts Institute of Technology, August 1988

[66] Shrobe H.E. , *Constraint Propagation in VLSI Design: Daedallus and Beyond*, Massachusetts Institute of Technology, Abstracts from Spring 1980 MIT VLSI Research Review, 1980

[67] Stallman R.M. , *Forward Reasoning and Dependency-Directed Backtracking in a System for Computer-Aided Design*, Artificial Intelligence 9, pp.135-196, 1977

[68] Steele G.L.Jr. , *The Definition and Implementation of a Computer Programming Language Based on Constraints*, AI-TR-595, Artificial Intelligence Laboratory, Massachusetts Instute of Technology, 1980

[69] Sussman J.G. and Steele G.L.Jr. , *Constraints - A Language for Expressing Almost-hierarchical Descriptions*, Artificial Intelligence 14, 1980

[70] Thinking Machine Corporation , *Introduction to Data Level Parallelism*, Technical report Series TR86-14, April 1986

[71] Traub K.R. , *A Dataflow Compiler Substrate*, Memo 261, Computation Structure Group, Laboratory for Computer Science, Massachusetts Institute of Technology, March 1986

[72] Traub K.R. , *A Compiler for the MIT Tagged-Token Dataflow Architecture*, Technical Report TR-370, Laboratory for Computer Science, Massachusetts Institute of Technology, August 1986

[73] Valdés-Pérez R.E. , *Spatio-Temporal Reasoning and Linear Inequalities*, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1986

[74] Vere S.A. , *Planning in Time: Windows and Durations for Activities and Goals*, IEEE Transactions on Pattern Analysis and Machine Intelligence, 1983

[75] Wesson R.B. , *Planning in the World of Air Traffic Control*, Proceedings of IJCAI, 1977

[76] Yuba T. , Shimada T. , Hiraki K. and Kanshiwagi H. , *Sigma 1: A Dataflow Computer for Scientific Computation*, Technical Report, Electrotechnical Laboratory, 1-1-4 Umescono, Sakuramura, Niiharigum, Ibaraki 305, Japan, 1984