MASSACHUSETTS INSTITUTE OF TECHNOLOGY
ARTIFICIAL INTELLIGENCE LABORATORY

A.I. Memo No. 1286 April 1991

Phase Space Navigator: Towards Automating Control Synthesis in Phase Spaces for Nonlinear Control Systems

Feng Zhao

Abstract

We develop a novel autonomous control synthesis strategy called Phase Space Navigator for nonlinear control systems, with which a controller for a nonlinear system can be automatically synthesized in phase spaces. The Phase Space Navigator generates control laws by synthesizing flow shapes of dynamical systems and planning and navigating system trajectories in the phase spaces. More specifically, the control synthesis strategy consists of a global control path planner, a local trajectory generator, and a reference trajectory follower. The global path planner finds optimal paths from an initial state to the goal state in the phase space, consisting of a sequence of path segments connected at intermediate points where the control parameter changes. A brute-force, fine-grain search in high-dimensional phase spaces would be prohibitively expensive. Modeling and parsing phase spaces into trajectory flow pipes provide a way to efficiently reason about the phase space structures and search for global control paths. The local trajectory generator uses the flow information about the phase space trajectories to produce smoothed trajectories. The trajectory follower tracks the planned reference trajectory, reactively corrects deviations, and resynthesizes the reference trajectory if the dynamics of the system changes significantly.

We have demonstrated the strategy with a program that automatically synthesizes global control paths for stabilizing a steel column buckling under compression. The Phase Space Navigator is particularly suitable for synthesizing high-performance control systems that do not lend themselves to traditional design and analysis techniques. It can also assist control engineers in exploring much larger design spaces than otherwise possible.

Keywords: automated control synthesis and analysis, intelligent control, nonlinear control, knowledge-based systems, planning and simulation, autonomous systems, dynamical systems, knowledge representation, numeric/symbolic processing.

This paper describes research done at the Artificial Intelligence Laboratory of the Massachusetts Institute of Technology, supported in part by the Advanced Research Projects Agency of the Department of Defense under Office of Naval Research contract N00014-89-J-3202, and in part by the National Science Foundation grant MIP-9001651. The author is also supported by a G.Y. Chu Fellowship.

Phase Space Navigator:
Towards Automating Control Synthesis in Phase Spaces for Nonlinear Control Systems*

Feng Zhao
MIT Artificial Intelligence Laboratory
545 Technology Square, Room 438
Cambridge, MA 02139

Abstract

We develop a novel autonomous control synthesis strategy called Phase Space Navigator for nonlinear control systems, with which a controller for a nonlinear system can be automatically synthesized in phase spaces. The Phase Space Navigator generates control laws by synthesizing flow shapes of dynamical systems and planning and navigating system trajectories in the phase spaces. More specifically, the control synthesis strategy consists of a global control path planner, a local trajectory generator, and a reference trajectory follower. The global path planner finds optimal paths from an initial state to the goal state in the phase space, consisting of a sequence of path segments connected at intermediate points where the control parameter changes. A brute-force, fine-grain search in high-dimensional phase spaces would be prohibitively expensive. Modeling and parsing phase spaces into trajectory flow pipes provide a way to efficiently reason about the phase space structures and search for global control paths. The local trajectory generator uses the flow information about the phase space trajectories to produce smoothed trajectories. The trajectory follower tracks the planned reference trajectory, reactively corrects deviations, and resynthesizes the reference trajectory if the dynamics of the system changes significantly.

We have demonstrated the strategy with a program that automatically synthesizes global control paths for stabilizing a steel column buckling under compression. The Phase Space Navigator is particularly suitable for synthesizing high-performance control systems that do not lend themselves to traditional design and analysis techniques. It can also assist control engineers in exploring much larger design spaces than otherwise possible.

*This paper is to be presented at the Third IFAC International Workshop on Artificial Intelligence in Real Time Control, sponsored by International Federation of Automatic Control (IFAC), California, September 1991.

1 Introduction

We develop a novel autonomous control synthesis strategy called Phase Space Navigator for nonlinear control systems, with which a controller for a nonlinear system can be automatically synthesized in phase spaces. The Phase Space Navigator generates control laws by synthesizing flow shapes of dynamical systems and planning and navigating system trajectories in the phase spaces. It is particularly suitable for synthesizing high-performance control systems that do not lend themselves to traditional design and analysis techniques. It can also assist control engineers in exploring much larger design spaces than otherwise possible.

The control synthesis strategy consists of a global control path planner, a local trajectory generator, and a reference trajectory follower. The global path planner finds optimal paths from an initial state to the goal state in the phase space, consisting of a sequence of path segments connected at intermediate points where the control parameter changes. A brute-force, fine-grain search in high-dimensional phase spaces would be prohibitively expensive. High-level descriptions of the phase space and trajectory flows provide a way to efficiently reason about phase space structures and search for global control paths. The local trajectory generator uses the flow information about the phase space trajectories to produce smoothed trajectories. The trajectory follower tracks the planned reference trajectory, reactively corrects deviations, and resynthesizes reference trajectory if the dynamics of the system changes significantly. The control synthesis program employs various techniques such as phase space cut-and-paste operation, programmed damping and switching, and piecewise-linear approach [16].

The strategy relies on the phase space knowledge obtained from an autonomous phase space analysis and modeling program called MAPS[1] that extracts and represents qualitative phase space structures of nonlinear dynamical systems of any order [17]. It is made possible by the geometric and dynamical modeling of the phase space behaviors in so-called *flow pipes*. MAPS generates a high-level description about a dynamical system describing equilibrium points and limit cycles, geometries of stability regions and trajectory flows, and their spatial arrangements and interactions in the phase space in a relational graph. The high-level description is sensible to human beings and manipulable by other programs.

The phase space description can assist engineers in designing controllers for complex systems. We envision that a control engineer uses MAPS to observe *qualitative changes* of phase space structures when varying control parameters: (1) the birth, disappearance, and movement of equilibrium points and limit cycles; (2) the enlargement and shrinking of stability regions; and (3) other phase space qualitative changes. The engineer can visualize the qualitative phase portrait with graphics rendering tools, interactively edit the system to observe corresponding changes, and cut-and-paste together useful portions of phase portraits so that the

[1]MAPS stands for Modeler and Analyzer for Phase Spaces.

composite has the desired properties.

In this paper, we automate the above scenario with the Phase Space Navigator that autonomously synthesizes nonlinear controllers in phase spaces. More importantly, the method will be able to synthesize a nonlinear control system whose phase space is high dimensional and difficult to visualize, or on which the desired control properties are impossible to obtain with traditional design techniques. The *topological* and *dynamical* modeling of the phase space stability regions and trajectory flows forms the basis for our method.

2 Automatic Control Synthesis in Phase Spaces

The Phase Space Navigator synthesizes a control system from a geometric point of view in the phase space. The control of a dynamical system is interpreted as the "steering" of a state trajectory, emanating from some initial state, to the desired state by a control signal.

2.1 Intelligent Navigation in Phase Spaces

The control objective for a stabilization problem is to synthesize a control path so that the physical system is brought to the goal state along the prespecified path and stays there afterwards under certain control. We are particularly interested in cases in which physical plants to be controlled operate in large regions where nonlinearities of the plants can not be ignored. When no global stabilization control laws can be found for the desired state with traditional control techniques, composite control paths have to be synthesized. Geometrically speaking, this is the case where initial states of the systems are far away from the desired state. Phase Space Navigator takes advantage of the underlying dynamics of the phase space flows to plan the trajectory both locally and globally and switches control at carefully planned time instances and places [16]. This type of control via phase space path planning dictates little control authority and achieves the desired control properties otherwise impossible to obtain or difficult to manually synthesize. It is a small, opportunistic dynamical alteration based on the global knowledge of phase space structures. As soon as the system enters the neighborhood of the desired state, a local linear controller stabilizes the system at the desired location.

2.2 Parsing Phase Spaces into Flow Pipes

Many nonlinear control systems do not have closed, analytic form solutions. Computational exploration is often the most common means for designing and analyzing global nonlinear controllers. Since a phase space is a union of an infinite number of trajectories, an exhaustive search for plausible control paths in the space of all the trajectories is computationally too expensive and numerically too sensitive to

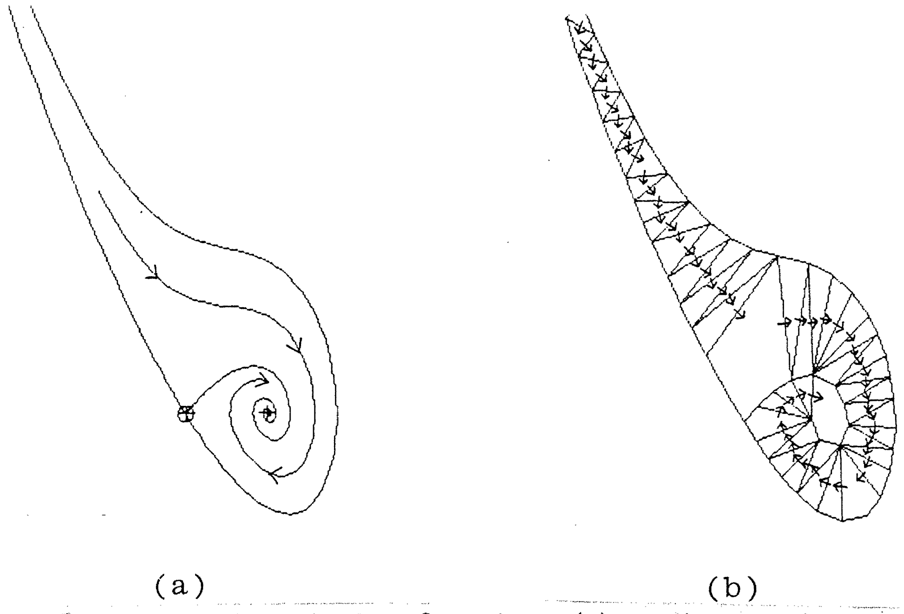


Figure 1: Grouping trajectories into flow pipes: (a) a collection of trajectories with the same qualitative feature; (b) a corresponding flow pipe.

uncertainties. We introduce the novel idea of flow pipes to model trajectory flows in a phase space.

The trajectory flows are modeled with flow pipes that describe the shape and direction of the flows. (The construction of the flow pipes is to be discussed in Section 4.) Figure 1(a) shows a portion of the phase space for the Lienard equation [16], consisting of all trajectories ending at the attractor denoted by the symbol $+$. A flow pipe groups this collection of trajectories that exhibit the same qualitative feature into an equivalence class, as shown in Figure 1(b).

The Phase Space Navigator searches in a much smaller space of equivalence classes of trajectories, the flow pipes. The expensive fine-grain search in phase spaces, possibly of high dimensions, is avoided. Furthermore, the flow pipe is a natural representation with respect to robustness to the effect of noises and uncertainties, compared with individual trajectories. Often, the synthesized reference trajectory has to be modified to accommodate uncertainties in the model. The flow pipe provides room for a small deformation of the reference trajectory within the flow pipe at runtime.

2.3 Two-Stage Automatic Control Synthesis

The Phase Space Navigator has two main modules: the planning module and the tracking module. The planning module synthesizes a desired trajectory, the reference trajectory, in the phase space based on the dynamics of the nominal model. The tracking module follows the reference trajectory and reactively corrects

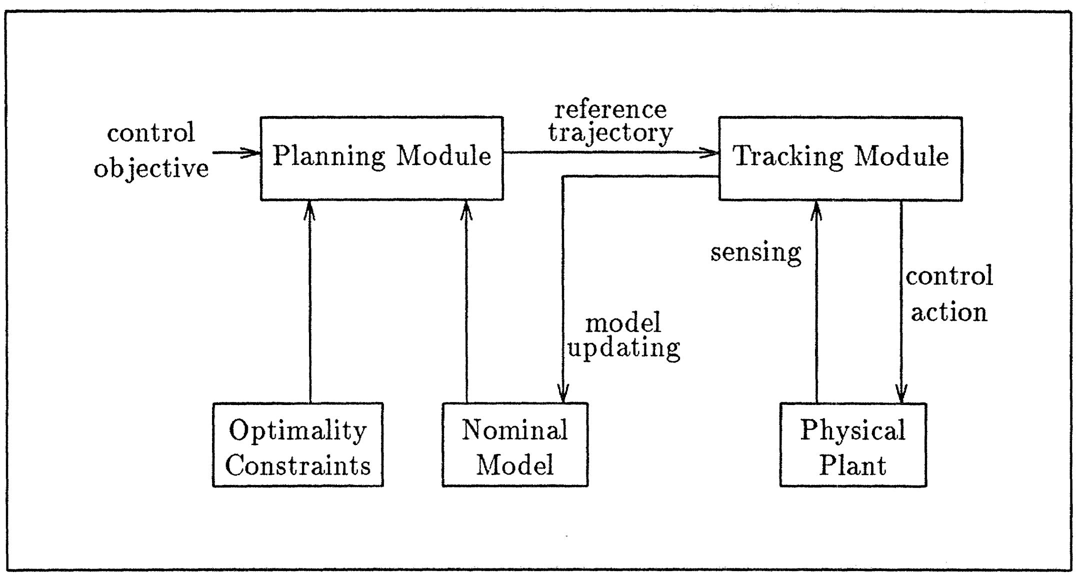


Figure 2: The Phase Space Navigator

local deviations. The parameters of the nominal model are estimated at runtime and the model is updated. When the dynamics changes significantly, the reference trajectory is resynthesized. Figure 2 shows the interplay between the two modules in the context of controlling a real physical plant.

2.3.1 Reference trajectory generation

The planning module consists of a global path planner and a local trajectory generator. The global path planner finds coarse global paths from the initial state to the goal state, utilizing the flow pipes of trajectories. The local trajectory generator fits smooth trajectory segments into the global path by slicing out trajectory segments from flow pipes or through local trajectory deformation. Figure 3 illustrates the data flows among components of the planning module.

(1) Global Path Planner

The global path planner searches for optimal paths from an initial state to a goal state in the phase space, subject to design constraints such as fast convergence and little overshooting. The high-level qualitative description of the phase space guides the search.

To synthesize a global path, MAPS first explores in a set of phase portraits indexed by the selected values of the control signal. It generates a sequence of phase space descriptions summarized in a geometric vocabulary. Flow pipes are then extracted from the descriptions. The global planner searches for a pipe path — a sequence of flow pipes interconnected at intermediate points — from the initial state to the goal state in this collection of flow pipes.

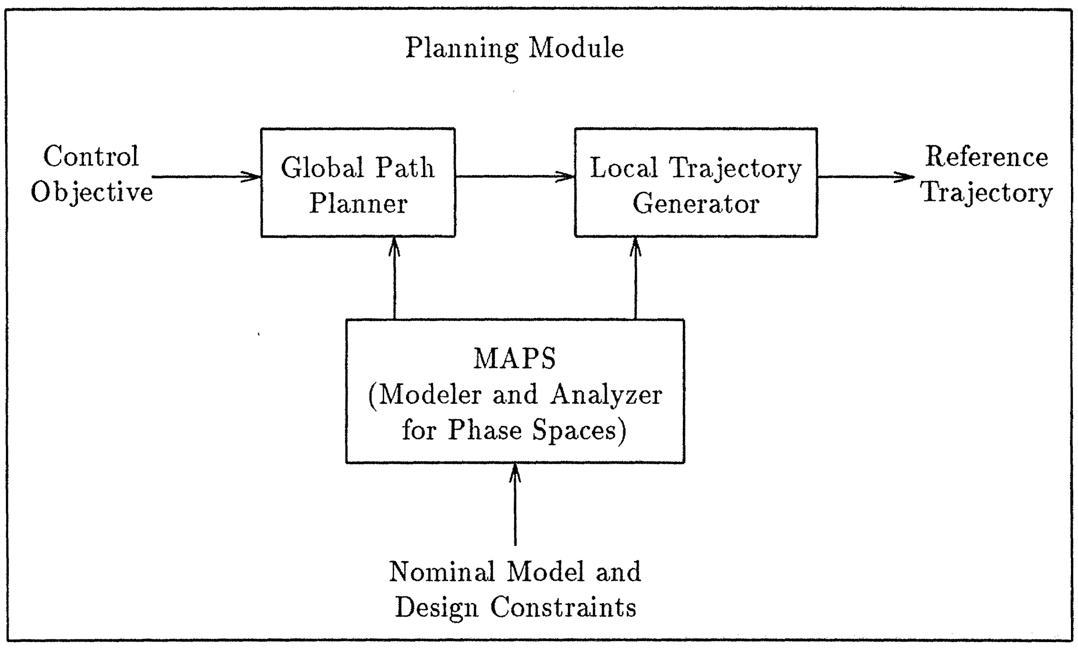

Planning Module
Control Objective
Global Path Planner
Local Trajectory Generator
Reference Trajectory
MAPS (Modeler and Analyzer for Phase Spaces)
Nominal Model and Design Constraints

Figure 3: Reference trajectory generation

For stabilization problems, we need to synthesize an attractor at the goal state. When the system enters the neighborhood of the goal state, a linear feedback controller is switched in to stabilize the system at the goal state.

(2) Local Trajectory Generator
The local trajectory generator produces smoothed trajectories connecting intermediate points. More specifically, trajectory segments are extracted from the flow pipes forming the global path. A trajectory segment is extracted from a flow pipe and is continuously deformed to connect intermediate points in the path. The local trajectory generator also smoothes out sharp interconnections of trajectory segments to reduce undesired transients, with local linear controllers or local sliding surface insertion [14]. The noises and parametric uncertainties of the synthesized reference trajectory are modeled with a thickened trajectory. To guarantee robustness the thickened trajectory segment is restricted within the flow pipe that contains the original reference trajectory segment.

The planning module generates a plan, i.e., the reference trajectory along with the control action for each segment of the reference trajectory.

2.3.2 Reference trajectory tracking

The tracking module follows the planned trajectory and reactively corrects the deviation due to uncertainties in dynamics modeling, disturbances, etc. The tracking is usually local to the region of the phase space containing the trajectory segment;

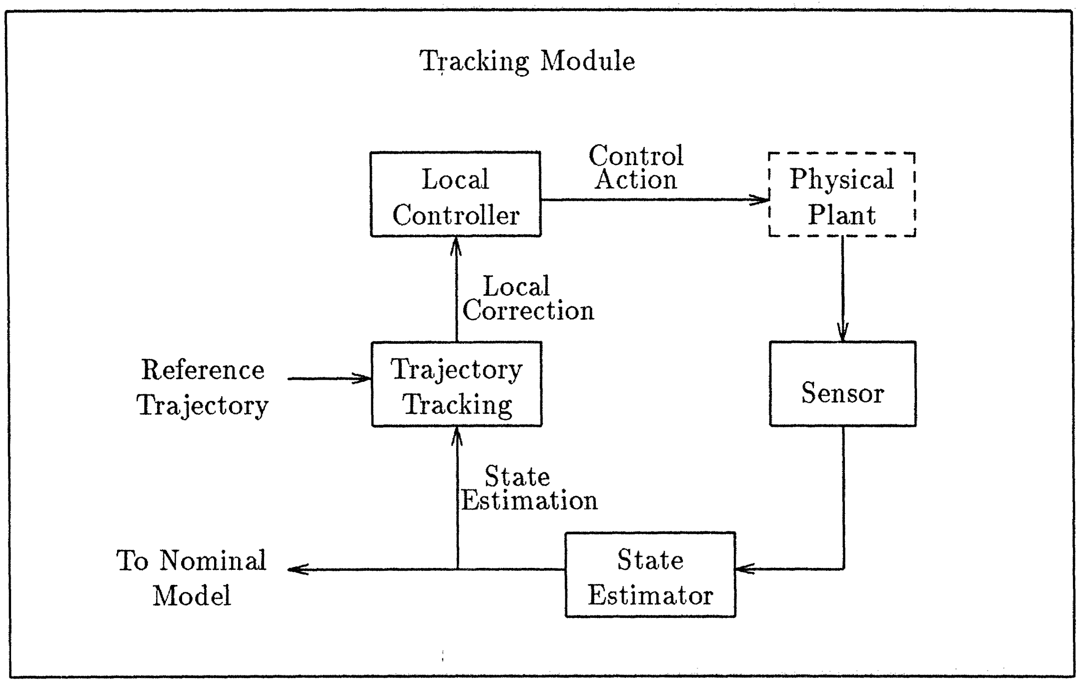


Figure 4: Reference trajectory tracking

see Figure 4. The planned reference trajectory provides the global feedforward reference term. The sensor and estimator measure the actual state, whose difference from the reference term is the local feedback correction term.

(1) Trajectory tracking and reactive correction

The Phase Space Navigator tracks a synthesized reference trajectory by sensing the state of the system, comparing the state with the reference trajectory, and computing the current control action. If the system stays within the tolerance of the reference trajectory, the controller simply looks up for a control action in the planned trajectory. If the system deviates from the desired trajectory, a local linear controller is switched in to force the system back on track. The trajectory correction is local and reactive. It is designed to ensure the robustness of the system in the presence of small changes in system parameters and noises in measurements.

(2) Model updating and reference trajectory resynthesis

The tracking module estimates the model parameters through the observables and updates the nominal model. When the dynamics of the system changes substantially such that the planned global path is no longer feasible, the Phase Space Navigator calls the planning module to synthesize a new reference trajectory.

3 The Automated Control Synthesis Algorithm

We consider a control system $x' = f(x, u)$, where $x \in R^n$ is the state variable and $u \in R^m$ is the control parameter. The initial state of the system is x_s and the goal state for the system to reach is x_g. The set of admissible values for the control parameter u is $\{u_i\}$. The planning algorithm searches for paths from x_s to x_g, subject to certain design constraints, in the phase spaces indexed by control parameter values $\{u_i\}$. The tracking algorithm follows the synthesized reference path. Figure 5 shows the flow chart of the planning algorithm.

3.1 The Trajectory Planning Algorithm

1. **generating qualitative phase space descriptions and flow pipes:**

   Qualitatively model the phase space structure and flow pipes for each of the phase portraits indexed by $\{u_i\}$.

2. **constructing flow pipe graph:**

   Paste together flow pipes from the phase portraits and form a directed graph, the flow pipe graph, where nodes are the intersections of flow pipes and edges are the segments of flow pipes. In particular, x_s and x_g are nodes in the graph.

3. **weighing flow pipe graph:**

   Compute weight for each edge, a segment of a flow pipe, according to traveling time, smoothness, etc. (The weight can be a single value or a range of values.)

4. **clustering path components:**

   Cluster the flow pipe graph into flow pipe path components. If x_s and x_g are in the same flow pipe path component, go to step 7. Otherwise, the goal is unreachable with the given set of parameter values $\{u_i\}$; continue.

5. **searching for partial paths from x_s to x_g:**

   Compute the reachable set R_{x_s} from x_s and the reverse reachable set R_{x_g} from x_g in the flow pipe graph. Denote the gap between R_{x_s} and R_{x_g} to be $G = \|R_{x_s} - R_{x_g}\|$.

6. **tuning control parameter:**

   Search for a value of the control parameter u outside $\{u_i\}$ such that the corresponding phase portrait has flow pipes, when added to the flow pipe graph, reducing the value of G. If G is zero, i.e., the gap between R_{x_s} and R_{x_g} is bridged, collect the flow pipe paths from x_s to x_g and go to step 8. Otherwise, repeat step 6 until no more parameter values to search, in which case return the collection of partial flow pipe paths from x_s to x_g.

7. **finding paths from x_s to x_g:**

   If looking for the shortest paths from x_s to x_g, run a standard shortest path algorithm on the graph. If searching for all paths p from x_s to x_g subject to the constraint $C(p) \leq C$, use the following algorithm.

```
(a) $P_{global-paths} = \emptyset$;
(b) $P_{partial-paths} = \{(x_s, x_s)\}$;
(c) **for** each path $p_i \in P_{partial-paths}$, ending at $x_i$, **do**
        $P_{p_i} = \{p_i * (x_i, x) | x$ one edge reachable from $x_i$, $x \neq x_g$,
              $C(p_i) + W(x_i, x) \leq C\}$;
        $P'_{p_i} = \{p_i * (x_i, x) | x$ one edge reachable from $x_i$, $x = x_g$,
              $C(p_i) + W(x_i, x) \leq C\}$;
        $P_{global-paths} = P_{global-paths} \cup P'_{p_i}$;
        $P_{partial-paths} = (P_{partial-paths} - p_i) \cup P_{p_i}$.
(d) if $P_{partial-paths} \neq \emptyset$, goto (7c);
    otherwise, continue.
(e) if $P_{global-paths} = \emptyset$, goto step 5;
    otherwise, continue.
(f) order paths in $P_{global-paths}$ according to their costs and return.
```

   Note: $C(p_i)$ is the cost of a path p_i. $W(x_i, x_j)$ is the weight of the edge (x_i, x_j). "$*$" is the path composition operator.

8. **generating piecewise trajectories:**

   Select a representative trajectory segment from each flow pipe forming the flow pipe path from x_s to x_g and paste them together at intersections of flow pipes.

9. **smoothing trajectories:**

   Smooth each intersection of trajectory segments through trajectory homotopy deformation. Order the smoothed trajectories with costs and return.

3.2 The Trajectory Tracking Algorithm

1. **sensing and estimating the physical system:**

   Sense the state x of the physical system, estimate the model parameters, and update the nominal model of the system. If x is in the neighborhood of the goal x_g, go to step 5. If the nominal model changes substantially, signal and go to step 4. Otherwise, continue.

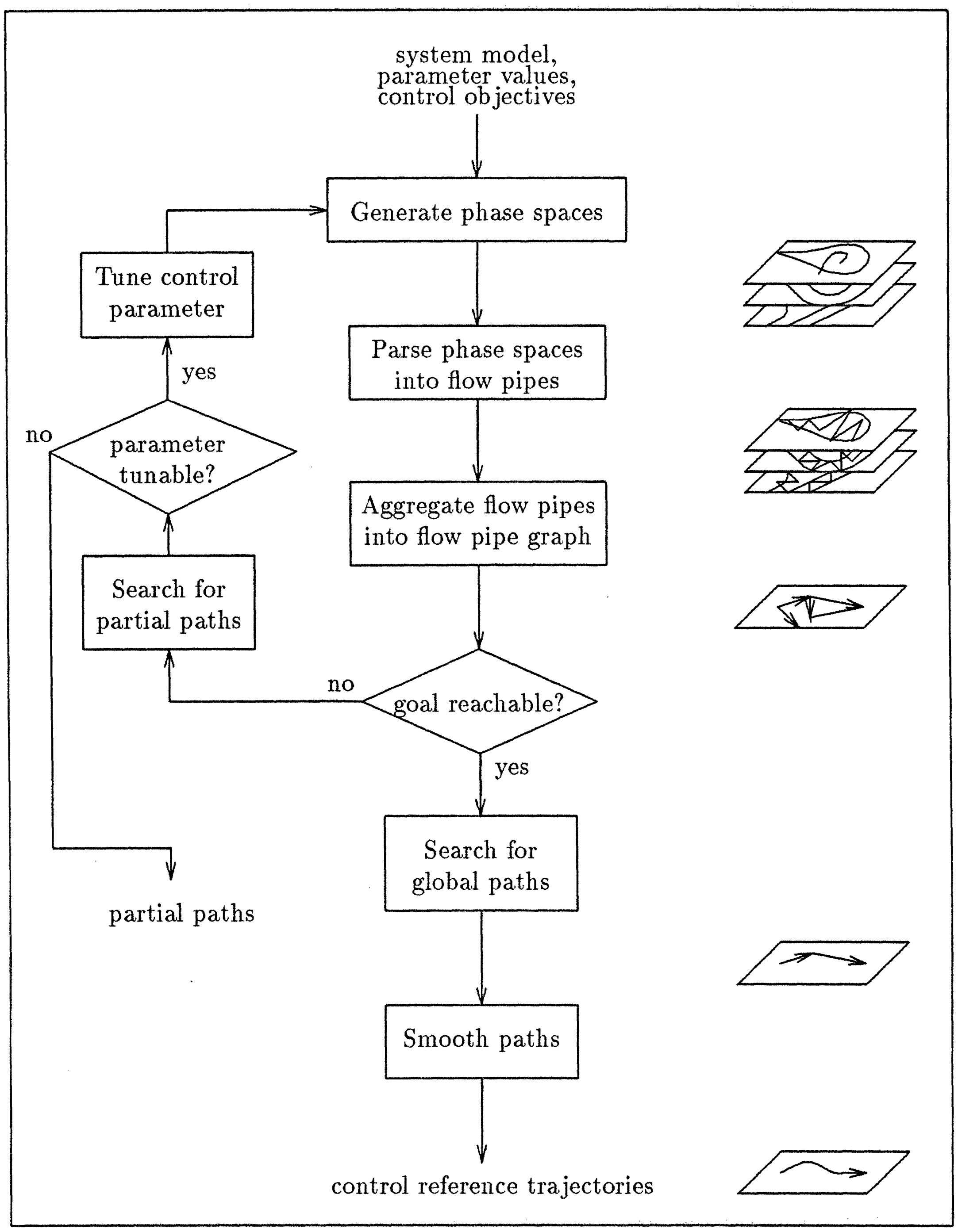


Figure 5: The flow chart of the trajectory planning algorithm

2. **computing control action:**

   Based on the synthesized reference trajectory and the observation from step 1, compute control tracking term u_{track} and correction term $u_{correct}$. If $u_{correct}$ is unobtainable with a local linear correction, check alternative paths to the goal from the collection of suboptimal paths. If the goal is unreachable, go to step 4. Otherwise, the control action is $u = u_{track} + u_{correct}$; continue.

3. **generating control action:**

   Tune the control parameter to u_{track} and generate $u_{correct}$ with a local linear feedback controller. Drive actuators and go to step 1.

4. **resynthesizing reference trajectory:**

   Call the planning module to resynthesize a reference trajectory and go to step 2.

5. **goal stabilizer:**

   Stabilize the system at x_g with a local linear controller, subject to uncertainties and noises.

3.3 Discussion on the Algorithm

1. On-line and off-line synthesis:

   The planning module generates smoothed reference trajectories in the phase space. The constraints on response time and the availability of computational resources dictate whether the computation of planning is done on-line or off-line. If the planning is done off-line, the synthesized plan is compiled into a table. At the runtime the controller performs a table lookup for a control action. On-line reference trajectory synthesis and tracking require substantial computational power. The Supercomputer Toolkit [4] provides an ideal platform for experimenting with the above control synthesis algorithm on real-time control applications.

   In step 1 of the planning algorithm, the phase space structure and flow descriptions can be either computed once for all the control parameter values or generated on demand in the search for optimal paths. The later case will be more suitable for on-line synthesis.

2. Search for optimal paths:

   We present the planning algorithm for a control design problem with one initial state and one goal state. The algorithm also applies to control systems of "one initial state/many goal states", "many initial states/one goal state", or "many initial states/many goal states". We note that the flow pipe graph is a directed graph where nodes are intersections of flow pipes and edges are segments of flow pipes. (The graph may contain cycles.) The search for optimal paths is formulated as a search for shortest paths in the directed graph.

For a "one initial state/one goal state" or "one initial state/many goal states" problem, the Dijkstra's algorithm for the single source shortest paths in a directed graph runs in $O(E^2)$ for a graph with V vertices and E edges [6].

A "many initial states/one goal state" problem can be converted to the "one initial state/many goal states" problem by reversing the directions of all edges; and a "many initial states/many goal states" problem is solved by the Floyd-Washall algorithm in $O(V^3)$ [6].

3. Suboptimal paths:

   The suboptimal paths can be useful when the optimal paths are no longer judged feasible. They can be stored in the table in addition to the optimal ones. A controller can opportunistically choose among available trajectory paths according to the desired properties at a control switching point.

4. discrete vs. continuous control parameter spaces:

   A version of the synthesis algorithm is presented for the control parameter initially taking values from a finite discrete set. An example of finite-value control systems is the satellite attitude controller, which stabilizes the antenna direction of the satellite subject to disturbances by turning on or off high thrust jets. Other such examples are switching power regulators.

   The method also applies to control systems with continuous control parameter spaces. The continuous parameter space is first sampled at discrete points. Then techniques such as those of Abelson's bifurcation interpreter [3] can be employed to search for distinct behaviors at finer scales.

4 Trajectory Flow Pipes

The flow pipes are further geometric and dynamical characterizations of stability regions of the phase space. They form the foundation for the search algorithm for synthesizing global paths.

A flow pipe models a collection of trajectories exhibiting the same qualitative features in the phase space. It is an equivalence class of trajectory paths, each of which can be continuously deformed to another one, delimited by the boundaries of stability regions and trajectories connecting equilibria[2].

[2]Formally, a flow pipe is a homotopy equivalence class of trajectory paths delimited by the boundaries of stability regions and trajectories connecting equilibria. If the ends of a flow pipe are squeezed to points, the flow pipe is a path homotopy equivalence class of trajectories.

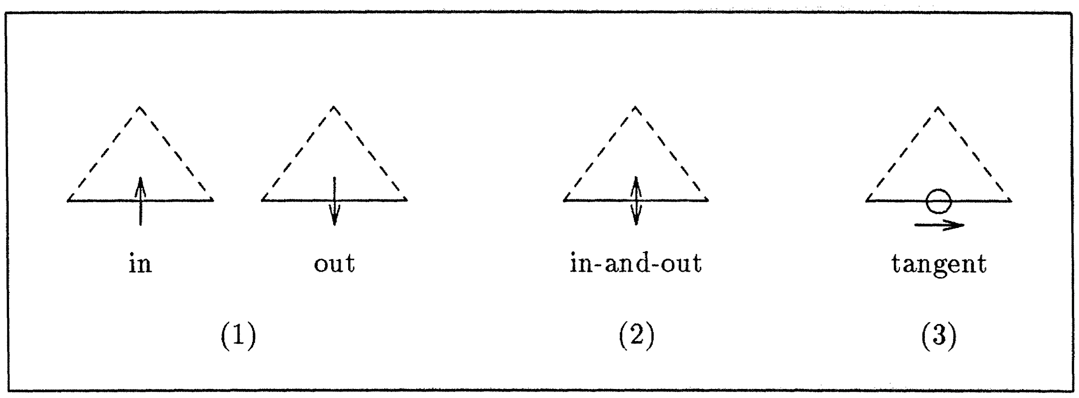


Figure 6: Classifying faces of simplices with respect to flows

Computationally, a flow pipe is constructed by aggregating consistent geometric pieces with respect to the way that the flow travels from piece to piece. The catalogue of flow types on boundaries of the geometric pieces is a set of consistency constraints derived from the underlying dynamics of the vector field.

4.1 Constructing flow pipes

MAPS tessellates a phase space with simplices and geometrically models the stability regions with polyhedra. Each proper face f of a simplex s is classified into one of the three types with respect to the direction of the flow (see Figure 6), according to the following labeling scheme:

1. **unidirectional flow:**
   label *in:* Trajectories flow into s on f;
   label *out:* Trajectories flow out of s on f.

2. **bidirectional flow:**
   label *in-and-out:* Trajectories flow into s on some portion of f and flow out of s on other portion.

3. **zero flow:**
   label *tangent:* The flow does not intersect f. Hence the proper face f is on the boundary of the flow.

A proper face of a simplex is called monotonic with respect to both the flow and the simplex, if the face is labeled either type 1 or type 3 in the above labeling scheme. A simplex is monotonic with respect to the flow if all of its proper faces are monotonic. Similarly, a polyhedron is monotonic if all of its proper faces are monotonic. Given two simplices agreeing on a non-monotonic proper face, the common non-monotonic face is canceled in the polyhedron formed by the two simplices; see Figure 7(a). The polyhedron continues to grow in this way until all of its proper faces are monotonic. The monotonic polyhedra are then aggregated together to form flow pipes in such a way as to conserve the flow on their proper faces; see Figure 7(b). The following algorithm groups simplices into smallest monotonic polyhedra and clusters monotonic polyhedra into flow pipes, the largest monotonic polyhedra.

Flow Pipe Construction Algorithm

1. label proper faces of each simplex with respect to the flow.
2. cluster simplices into monotonic polyhedra:
   Cluster simplices into equivalence classes with the equivalence relation on the proper faces of the simplices: two simplices are equal if they agree on a proper face with the same flow label: *in-and-out.*
3. cluster monotonic polyhedra into flow pipes:
   Cluster monotonic polyhedra into equivalence classes with the equivalence relation on the proper faces of the polyhedra: two monotonic polyhedra are equal if they agree on a proper face with consistent flow directions: *in* and *out*, respectively.
4. order the monotonic polyhedra in each class from step 3 according to the flow direction and form a flow pipe. return.

The resulting flow pipes model the trajectory flows. The boundary of a flow pipe is formed by the proper faces labeled *tangent* of simplices in the pipe.

A flow pipe contains time information of the trajectory flow. The monotonic polyhedral pieces in a flow pipe approximate successive flow wave fronts indexed by the time. The flow pipe also characterizes the Lyapunov exponent [8] of the flow with the expansion and contraction of a volume element, for example, the crossover section of the flow pipe, along the flow pipe. The progressive movement of the flow wave front and its change in shape quantitatively describe the flow.

4.2 Classifying shapes of flow pipes

Flow pipes capture *dynamical shapes.* A flow pipe ends at an attractor and/or starts from a repellor. Saddles can not be in a flow pipe except for the boundary of the pipe, since flow pipes are further decomposition of the stability regions. In a bounded region, each flow pipe either ends at an attractor or leaves the region.

Flow pipes exhibit *topological shapes.* Each flow pipe is classified according to its relative shape. A flow pipe around a stable limit cycle is a closed pipe. Most pipes are open. The open pipes can be relatively straight or highly wound. One, two, or more pipes can side by side wind into a spiral, called n-winding pipes. See Figure 8. We classify the pipes into the following categories:

1. open pipes: (a) straight pipes; (b) wound pipes.

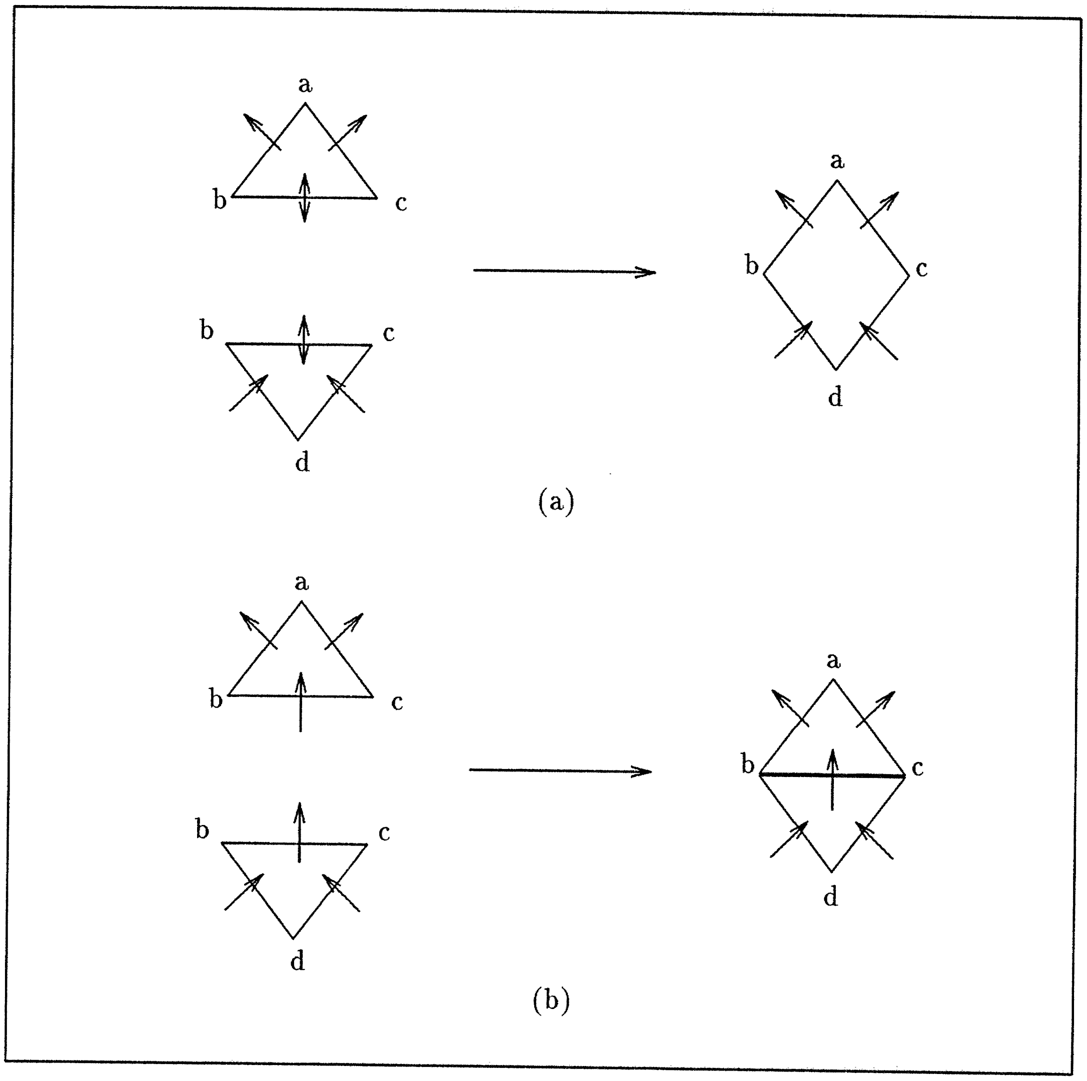


Figure 7: Construction of flow pipes: (a) grouping simplices to form a monotonic polyhedron by canceling common non-monotonic faces; (b) aggregating monotonic polyhedra to form a flow pipe according to flow directions at the boundaries.

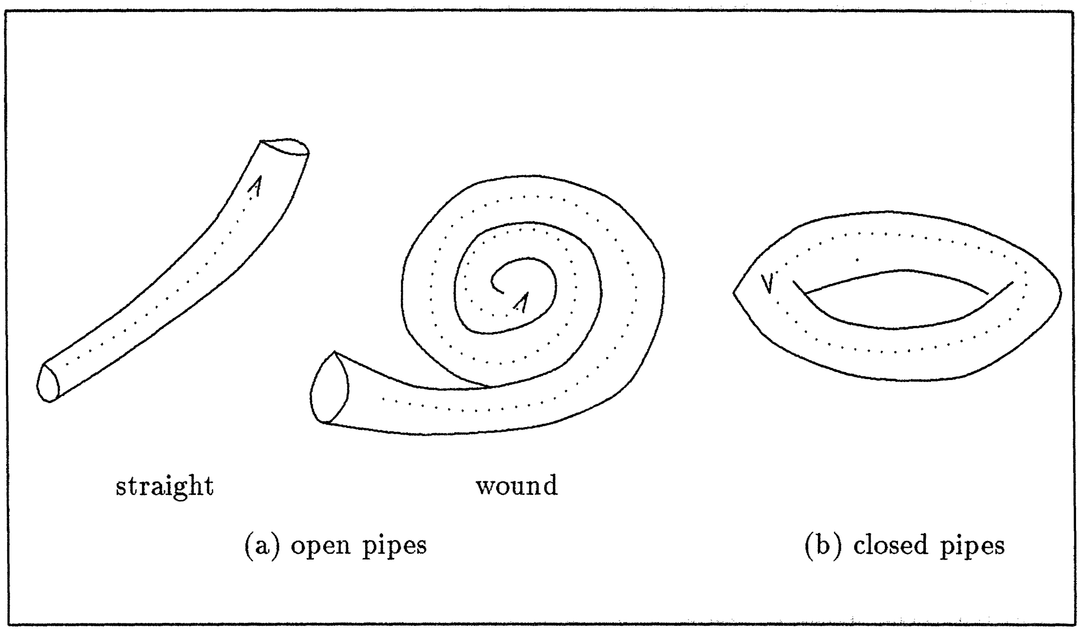


Figure 8: Topological shapes of flow pipes.

2. closed pipes.

Figure 1(b) of Section 2.2 shows a single pipe winds into a spiral shape at its end and forms the stability region for the attractor. More complicated patterns can occur with more pipes or in higher dimensions.

5 An Example: Stabilizing a Buckling Column

5.1 The column model

A thin elastic column, subject to axial end compressive loads, buckles around the principal axis, as shown in Figure 9. The nonlinearity is introduced by the nonlinear geometric stiffness of the column [11]. Stoker [15] gave a simplified model for the column subject to axial compressive force and viscous damping

$$mx'' + cx' + a_1 x + a_3 x^3 = 0, \tag{1}$$

where $a_1 = A + C - 2P/l$ and $a_3 = B + D - P/l^3$. The state x is the characteristic measure of the column deflection from the principal axis and x' is the velocity. The column has mass m and length $2l$. The axial load is P, and the coefficient of viscous damping is c. The bending stiffness is modeled by a primary hard spring with restoring force $Ax + Bx^3$ and a secondary hard spring with restoring force $Cx + Dx^3$. We rewrite equation (1) as a system of first-order equations

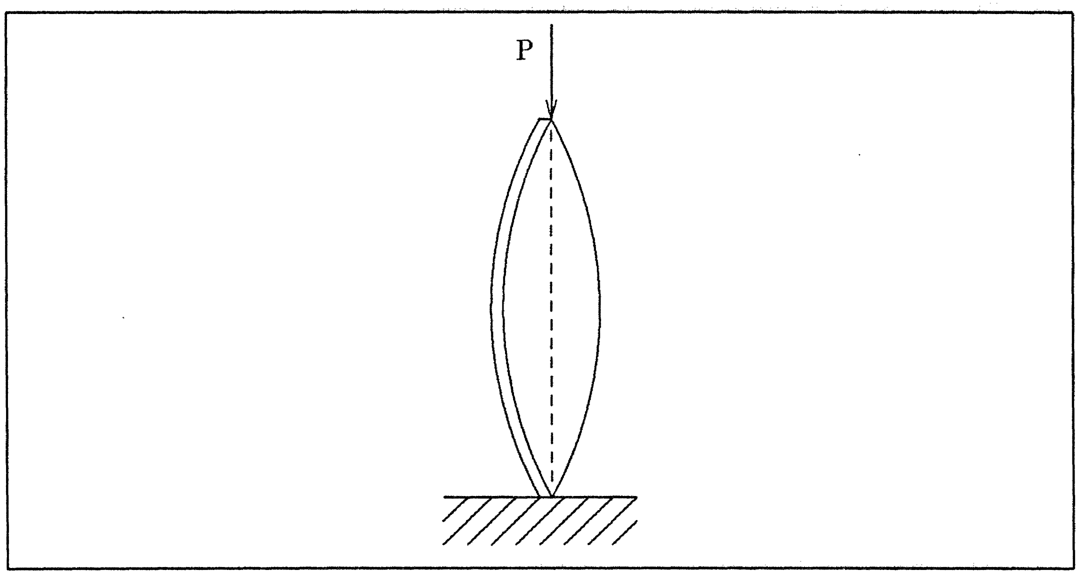


Figure 9: Buckling of a thin elastic steel column due to axial end loads.

$$\begin{cases} x_1' = x_2 \\ x_2' = \frac{1}{m}(-a_1 x_1 - a_3 x_1^3 - c x_2), \end{cases} \tag{2}$$

where x_1 represents the deflection and x_2 represents the velocity.

The system (2) describes the buckling motion of the column and represents only a single mode of vibration. For a long and slender column, vibrations are observed to occur primarily in the first mode [12].

When the axial load P is less than the critical load $P_{critical}$, the column oscillates around the principal axis and returns to the vertical state. It has only one stable state corresponding to the attractor at the origin of the phase space. Under a heavier load, i.e., $P > P_{critical}$, the column buckles to one side or the other. The phase space has a saddle at the origin and two attractors symmetrically arranged about the saddle; see Figure 10. The positions of the two buckled states depend on the external load P, the stiffness of the column, and the length l. The larger the load is, the farther away the buckled states are from the principal axis. The column breaks when the buckling exceeds certain limit. As P surpasses $P_{critical}$, the column undergoes a pitchfork bifurcation of equilibria [8]: the attractor at the origin gives birth to a saddle at the origin and two attractors on two sides.

5.2 Extracting and representing qualitative phase space structure of the buckling column

MAPS automatically analyzes the column model in the phase space and extracts and represents the qualitative phase space structure [17]. For parameter values

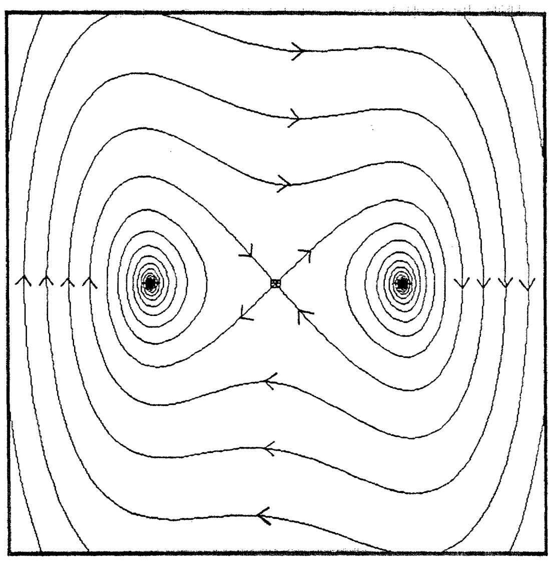


Figure 10: The phase space of a buckling column showing the stability boundaries and connecting trajectories. The horizontal axis x is the characteristic measure of the column displacement from its principal axis and the vertical axis x' is the velocity.

$a_1/m = -2.0$, $a_3/m = 1.0$, and $c/m = 0.2$ and phase space region $-3.0 \leq x_1 \leq 3.0$ and $-4.0 \leq x_2 \leq 4.0$, the program reports the following findings and represents them in a relational graph:

```
<equilibrium-points:
 1. attractor at:(1.41 0.)
 2. saddle at:(0. 0.)
 3. attractor at:(-1.41 0.)>

<relational-graph:
 1. stability-region for attractor at:*infinity*:
   stability-boundary:
   connecting-trajectories:
 2. stability-region for attractor at:(1.41 0.):
   stability-boundary:
     trajectory 1:(from *infinity* to (0. 0.))
     trajectory 2:(from *infinity* to (0. 0.))
   connecting-trajectories:
     trajectory 3:(from (0. 0.) to (1.41 0.))
 3. stability-region for attractor at:(-1.41 0.):
   stability-boundary:
     trajectory 1:(from *infinity* to (0. 0.))
     trajectory 2:(from *infinity* to (0. 0.))
   connecting-trajectories:
     trajectory 4:(from (0. 0.) to (-1.41 0.))>
```

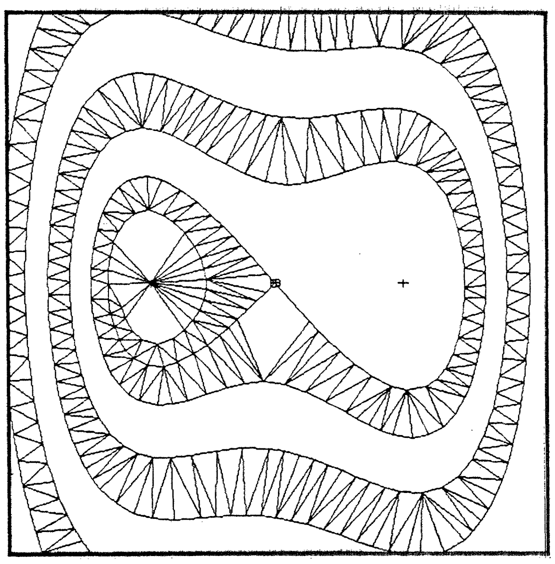


Figure 11: The flow pipe for the attractor on the left.

The program finds two attractors at (1.41, 0.0) and (−1.41, 0.0) and a saddle at the origin. It generates a high-level description of the phase space geometry: two banded stability regions associated with the two attractors, separated by the stable trajectories of the saddle at the origin. Figure 10 shows stability boundaries and connecting trajectories of two stability regions. Based on the geometric phase space representation, the phase space is further modeled with two flow pipes formed by aggregating geometric pieces, using the flow pipe construction algorithm. The pipe boundaries consist of separatrices of the two stability regions that approach the saddle and trajectories that connect equilibria. Figure 11 shows the flow pipe that ends at the left-hand attractor.

5.3 Synthesizing a control law for stabilizing the column in the phase space

We want to stabilize the column at its vertical state to support the axial end loads and to prevent it from breaking. Under sufficiently heavy load, the buckling motion of the column leads the column to one of the buckled states and, when the states are far away from the principal axis, induces the failure of the column.

We shall focus on global navigation in the phase space that synthesizes global reference trajectories leading to the desired goal state. Since the design of local linear controllers is relatively well understood, we shall not discuss the trade-offs among different designs for linear controllers and shall choose, for the purpose of demonstration, a simple linear feedback design. Local controllable regions of such linear controllers are quantified, given available control strength, and are used in constraining the design of global control paths.

The goal state of the control is the unbuckled state — the saddle at the origin of the phase space that does not have a stability region. We want to synthesize a non-zero stability region for the goal and maximize the region. The controlled column is of the form

$$\begin{cases} x_1' = x_2 \\ x_2' = \frac{1}{m}(-a_1 x_1 - a_3 x_1^3 - c x_2) + u, \end{cases} \tag{3}$$

where u is the control. In the model, mu has the same dimension as the force.

5.3.1 Synthesizing a local goal stabilizer

When the system gets close to the goal state, the Phase Space Navigator synthesizes a local linear controller, say a PID controller [7], to stabilize the system at the goal. The available control strength determines a neighborhood U_{ctr} of the goal within which the local linear controller suffices to stabilize the system at the goal.

Many mechanical systems employ force control. The velocity v is directly controllable, whereas the position x is indirectly controllable via $x' = v$. For example, the column could be stabilized against buckling with a force exerted perpendicular to the principal axis near the center of the column, as currently being investigated by Andrew Berlin. The controllable subspace of the phase space of the column is in the direction of x_2. Since the unstable subspace of the saddle intersects transversally the controllable subspace in a linear neighborhood of the saddle, the goal is locally stabilizable [10].

The local linear controller at the goal is specified as

$$u_{local-linear} = -k_1 x_1 - k_2 x_2, \tag{4}$$

where $k_1, k_2 \geq 0$ are feedback strengths. The values for k_1, k_2 are chosen according to the available control authority and the condition that the line $u_{local-linear} = 0$ is approximately in the unstable direction of the saddle in its neighborhood. For our purpose, we simply let $k_1 = 4.18$, $k_2 = 2.75$. The control strength $|u_{local-linear}|$ is constrained within ϵ, i.e., $|u_{local-linear}| \leq \epsilon$. This constrains the controllable region U_{ctr} of the local linear controller

$$U_{ctr} \subseteq \{(x_1, x_2) | \epsilon \leq k_1 x_1 + k_2 x_2 \leq \epsilon\}.$$

The local linear region around the saddle is also determined, within which local stable and unstable eigenvectors are good approximation for stable and unstable trajectories of the saddle. The curvature of the stable and unstable trajectories characterizes the size of the local linear region. When U_{ctr} is within the linear region, the above synthesized linear feedback controller suffices to stabilize the system at the goal.

5.3.2 Bending flow pipes to deform trajectories

The Phase Space Navigator automatically synthesizes a smooth global trajectory from an initial state x_s to the goal state x_g. We consider two cases for the initial states x_s.

(a) The column is initially buckling with sufficient velocity

The initial state x_s is far away from the saddle x_g in the phase space in this case. The search for global control paths to the goal is simple. The single flow pipe containing x_s intersects the local controllable region U_{ctr} of x_g. To synthesize a trajectory that enters U_{ctr}, the region U_{ctr} is backward projected in flow pipes to form a goal projection. Then the trajectory emanating from x_s is deformed towards the goal projection. The programmed deformation is designed to push the trajectory towards the nearest goal projection in the controllable direction x_2. Since the control is more effective when the direction of the vector field is relatively orthogonal to the direction of the controllable direction, switching points are inserted to turn off the controller when the angle between the two directions are below some threshold. Figure 12(a) shows, for example, a synthesized reference trajectory originating at $(-1, -3)$. The circles indicate places where the control of deformation switches. The flow pipe that leads to the attractor on the left contains the initial state. The goal is on the boundary of the pipe. Given the maximum control strength for the linear goal stabilizer (4) in Section 5.3.1, the controllable region U_{ctr} is bounded by the region specified as

$$|4.18x_1 + 2.75x_2| \leq 0.2.$$

In this demonstration, we choose an over-conservative region for U_{ctr} determined by

$$U_{ctr} = \{(x_1, x_2) \mid \; |0.835x_1 + 0.550x_2| \leq 0.04, \; |0.797x_1 - 0.605x_2| \leq 0.04\}, \qquad (5)$$

in which the local linear approximation of the vector field is also valid. The U_{ctr}, formed by flow pipes too, intersects the flow pipe that ends at the left attractor, as illustrated in Figure 13. It is backward projected to form two thin pipes enclosing the boundaries of the flow pipe. The global trajectory emanating from $(-1, -3)$ is pushed towards the goal projection with small deformation that is less than 10% of the vector field strength at any state or 0.2 in the normalized unit, whichever is smaller. The position x and velocity $v = x'$ of the controlled column are plotted against the time t in Figure 14(a) and (b), respectively. The control u is shown in Figure 15.

When the goal is not reachable with the current deformation, the synthesis algorithm tries again with increased control strength if possible. The synthesized trajectories could be further optimized with variational techniques on the collection of trajectories within the flow pipe segments [5].

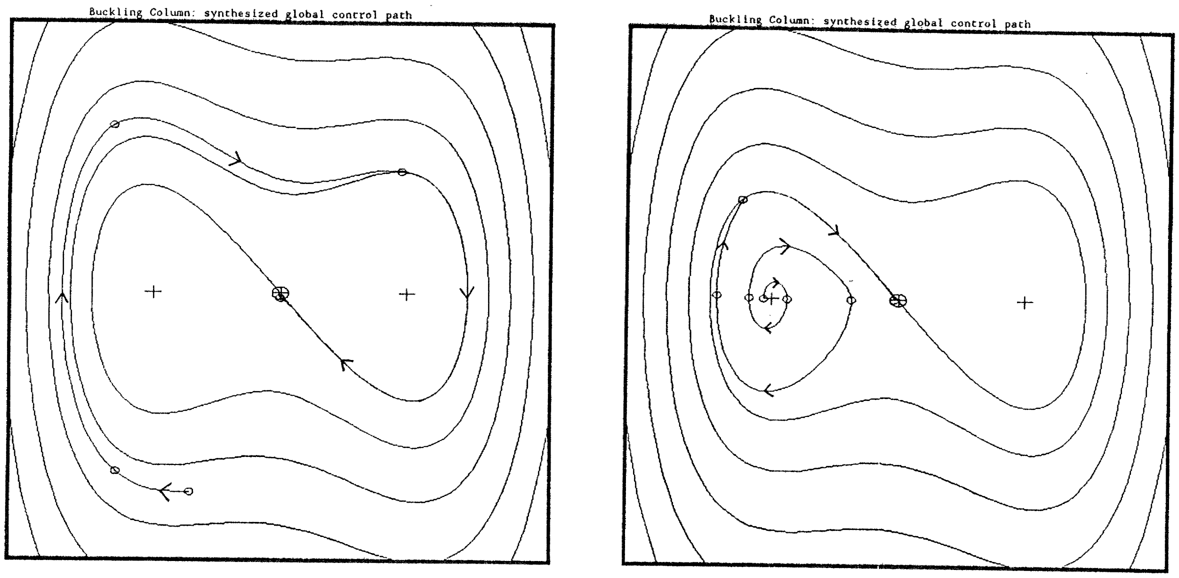


Figure 12: The synthesized control reference trajectories that lead to the unbuckled state corresponding to the saddle at the origin in the phase space of the column model: (a) starting with the column buckling with sufficient velocity; (b) starting with the column in the buckled state on the left.

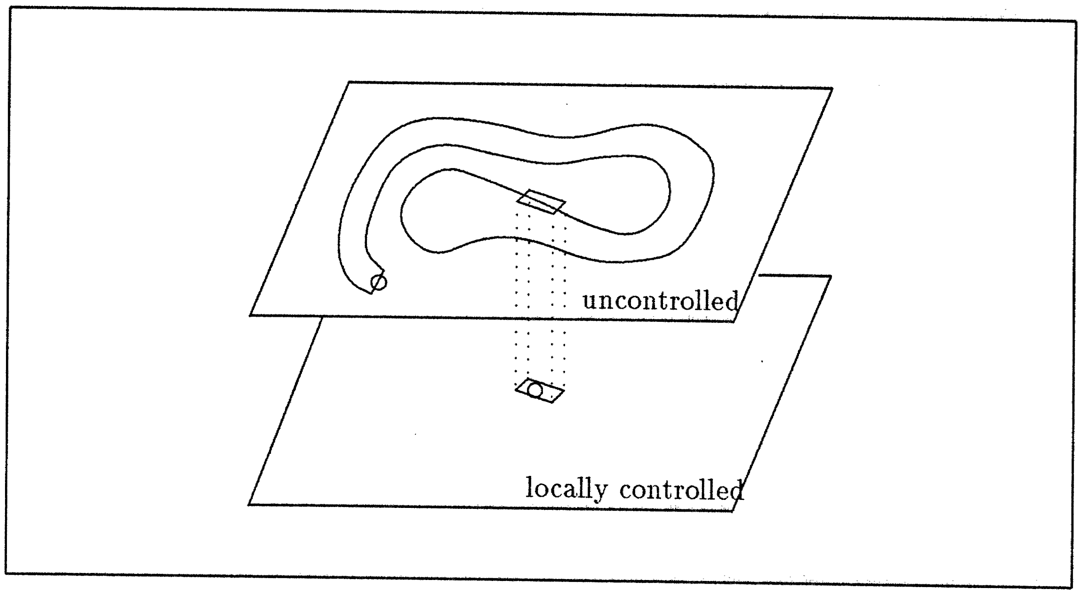
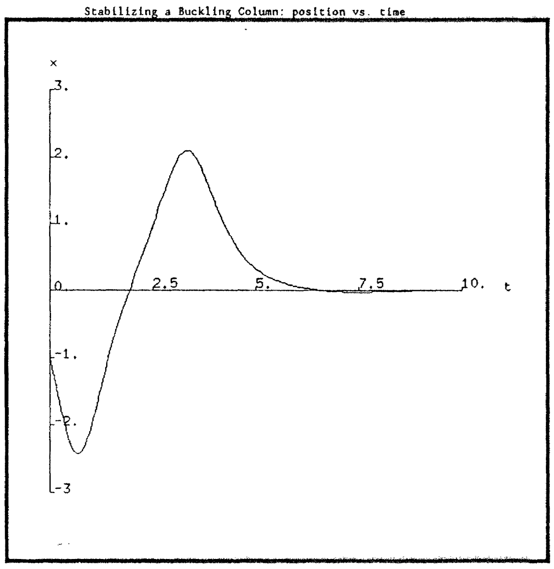
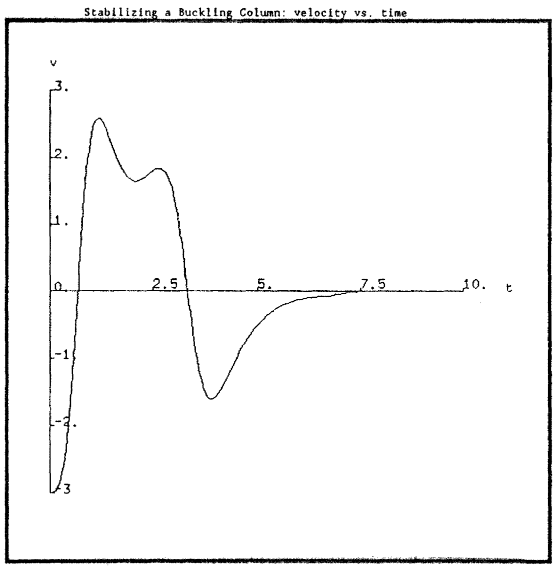


Figure 13: The portions of phase spaces for the uncontrolled column and the locally controlled column.





(a) (b)

Figure 14: Performance of the synthesized control reference trajectory for the column initially buckling with sufficient velocity: (a) position x of the column plotted against time t; (b) velocity $v(=x')$ of the column plotted against time t.

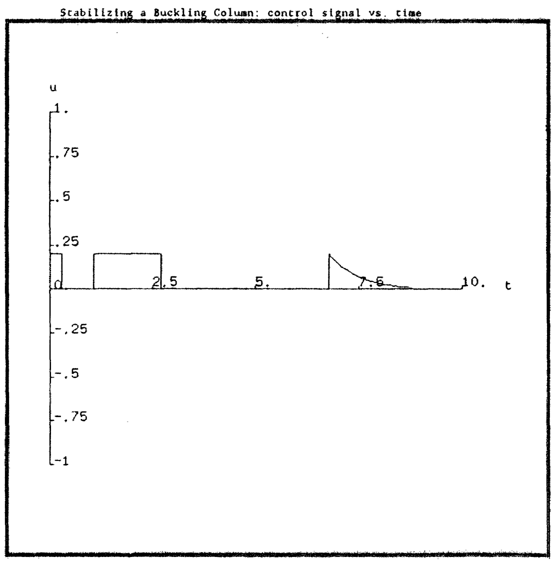
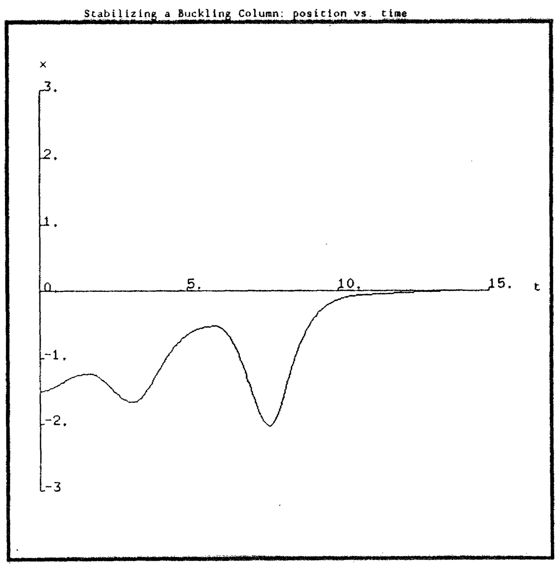
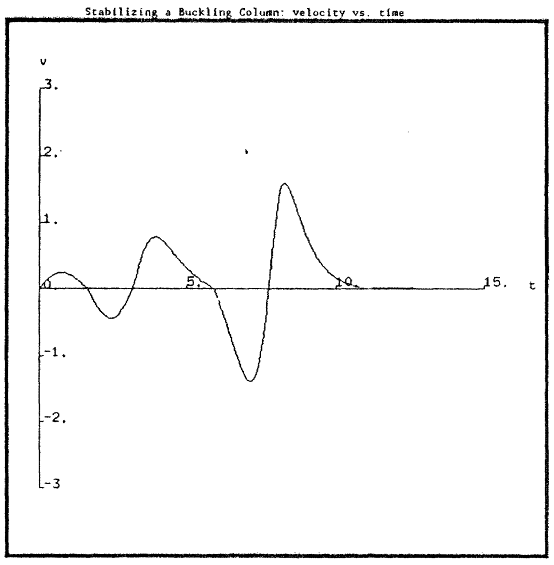


Figure 15: Control signal u for stabilizing the column initially buckling with sufficient velocity, plotted against time t.





(a) (b)

Figure 16: Performance of the synthesized control reference trajectory for the column starting at the buckled state: (a) position x of the column plotted against time t; (b) velocity $v(= x')$ of the column plotted against time t.

(b) The column is initially buckled

The control objective here is to pull the column out of the buckled state and bring it close to the unbuckled state. Since the flow pipe containing the initial state does not intersect U_{ctr}, a different control strategy must be employed. The program uses the following strategy to synthesize a control path.

The synthesized reference trajectory is shown in Figure 12(b). Again, the control parameter changes at places marked by circles. The column is initially near the buckled state at $(-1.41, 0.0)$. Its uncontrolled trajectory would spirally approach the attractor in the clockwise fashion. The control is exerted in such a way as to swing the trajectory away from the buckled state to approach the goal projection of the saddle. When the trajectory intersects the goal projection, the control is switched off so that the system slides along the uncontrolled trajectory. As soon as the system enters U_{ctr} determined by (5), the linear controller is switched in. The control strength is less than half of the vector field strength or 1.0 in the normalized unit, whichever is smaller. The position x and velocity $v = x'$ of the controlled column are plotted against the time t in Figure 16(a) and (b). The control u is shown in Figure 17.

Tracking the synthesized reference trajectory is executed in the same way that the deformation is done.

5.4 The phase space modeling makes the global navigation possible

The qualitative description of the phase space structure and the geometric modeling of the trajectory flows provide a "map" for navigating system trajectory to the goal in the phase space.

The directed graph constructed from the flow pipes is used in searching for global paths and in determining whether the goal is reachable. The flow pipes also make it possible to characterize the more microscopic spatial and temporal relations between the current state and the goal state. For example, the collection of simplices near the goal forms a neighborhood of the goal that determines whether or not the trajectory has missed the goal in the flow pipe. The deformation of global path segments is constrained by reasoning about the spatial relation with flow pipes.

5.5 Other control problems

The stabilization for a buckling column is closely related to the pole-balancing problem, where the control objective is to balance a pole at the vertically upward state. In the phase space of the pole model, the saddle is the upward state of the pole and the two attractors on two sides of the saddle correspond to the same downward state of the pole.

The control objective for pole-balancing is to bring the pole to the upward state from any other positions, even when the pole is initially hanging downward. The problem has been studied as a standard testbed for many control designs, most of which focus primarily on linear feedback control in linear regimes. With our phase space navigation strategy, a globally stable reference trajectory can be synthesized, similar to the one for the buckling column discussed earlier.

The optimal design of such trajectories has many practical implications. For example, the efficiency of cargo handling work at shipyards depends largely on the operation of overhead cranes [13]. Figure 18 illustrates such a crane driven by a trolley drive motor and a hoist motor. The bottleneck of the ship unloading operation is the transfer of cargo from a ship to waiting trucks; therefore, minimizing this transfer time brings about a large cost saving. The planar motion of the load hanging at the moving trolley can be modeled as a swinging pendulum with a moving fixed point and a hoisting cable. The trajectories of such overhead cranes could be optimally designed with the Phase Space Navigator. This would be an important step towards the full automation of cargo handling work without a crane operator.

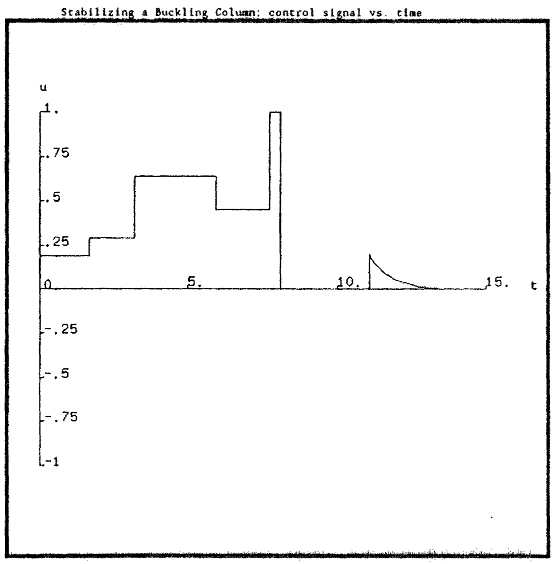


Figure 17: Control signal u for stabilizing the column starting at the buckled state, plotted against time t.

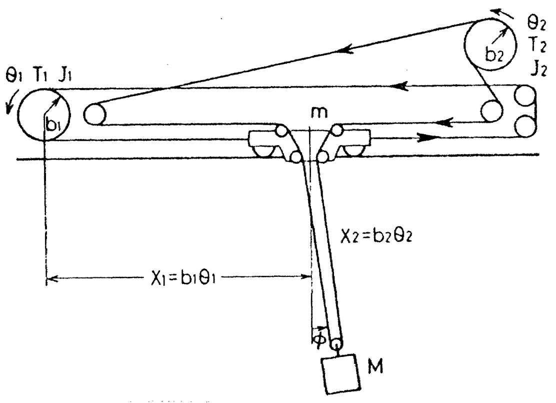


Figure 18: The mechanical model for an overhead crane unloading cargo from ships, reproduced from [Sakawa&Shindo, 1982]. The crane is equipped with a trolley drive motor and a hoist motor. The planar motion of the load is modeled as a swinging pendulum hanging at the moving trolley.

6 Related Work in Nonlinear Control

The gain scheduling and cellular control approaches in nonlinear control engineering are related to our phase space navigation method.

In gain scheduling [14], the system is linearized around a priori selected operating points. A linear controller is designed for each operating point. Interpolation is used in regions between the operating points. Gain scheduling is an open-loop adaptation where parameters of a regulator change in a prespecified way. Autopilots and chemical process control are instances of gain scheduling. The method requires human engineers to decompose a phase space into locally linear regions. Our method differs from the gain scheduling in that the phase space is decomposed automatically, even into global nonlinear ones, and the reference trajectory is synthesized automatically to navigate the system in the phase space based on the knowledge of the phase space structure.

The cellular control method is based on the cell state space concept [9]. It has its root in dynamic programming. The state space is first discretized into a cell space. The control law is synthesized with respect to some performance objective, such as minimal time. Then the control values at discrete states are programmed into a table. Run-time control is to perform a table-lookup. This method requires an exhaustive search for control paths at very fine grain. The cost in high dimensional state spaces would be prohibitive. The Phase Space Navigator, however, decomposes the phase space into a manageable collection of flow pipes and searches for global paths in this collection.

7 Conclusions

We have developed and demonstrated an automated control synthesis method, the Phase Space Navigator, for synthesizing controllers for nonlinear systems in the phase spaces. We have discussed primarily finite-value control systems as examples to illustrate the method and have noted that the method also applies to control systems with continuous parameter spaces.

The Phase Space Navigator synthesizes global reference trajectories using knowledge of phase space structures provided by the modeling and analysis program. A phase space is modeled with flow pipes that are collections of trajectories having the same qualitative behaviors. The flow pipes provide a way to efficiently search for and reason about global structures of phase spaces. The global nonlinear control synthesis becomes a graph search problem with this representation of the phase space. We have shown how global reference trajectories can be automatically designed with the example of stabilizing a buckling column.

With the novel idea of grouping an infinite number of trajectories into a manageable collection of flow pipes, the difficult task of synthesizing a nonlinear controller is formulated as a computational problem that requires a combination of substantial numerical, symbolic, and combinatorial computations and spatial reasoning techniques. Such a formulation provides a concrete substrate and a rich domain for exploring and developing new reasoning and representation techniques. These techniques would be otherwise hard to grasp in traditional artificial intelligence research.

The leverage of our automatic synthesis method comes from applications whose operating regions are grossly nonlinear and on which high-performance global controllers are impossible to synthesize with traditional techniques. Potential engineering applications include large flexible space structures, robot manipulator planning, satellite attitude control, switching power regulators, etc. The research in automated nonlinear control synthesis that actively exploits the knowledge of nonlinear dynamics in phase spaces has just begun. We have primarily focused on the automatic synthesis for global nonlinear controllers and have not addressed the important issues of modeling, sensing, estimation, and high-quality linear controller design. The exploration and development of a collection of new methods tackling these issues in all dimensions can revolutionize the synthesis and analysis of high-performance nonlinear control systems.

8 Acknowledgments

The author would like to thank Harold Abelson, Andrew Berlin, Martin Buhler, Daniel Frost, Weiping Li, Marc Raibert, Gerald Sussman for many helpful discussions and members of MIT Project on Mathematics and Computation for developing the MIT Scheme programming language and utilities. In particular, the author is indebted to Franklyn Turbak for his comments that greatly improved the content and presentation of this paper.

References

[1] H. Abelson and G. J. Sussman, "The Dynamicist's Workbench I: Automatic Preparation of Numerical Experiments." In: *Symbolic Computation: Applications to Scientific Computing.* R. Grossman (ed.), SIAM, Philadelphia, PA, 1989.

[2] H. Abelson, M. Eisenberg, M. Halfant, J. Katzenelson, E. Sacks, G.J. Sussman, J. Wisdom, and K. Yip, "Intelligence in Scientific Computing." *CACM*, 32(5), May 1989.

[3] H. Abelson, "Bifurcation Interpreter: A step towards the automatic analysis of dynamical systems." *Int. J. of Computers and Mathematics with Applications*, 20(8), June 1990.

[4] H. Abelson, A. Berlin, J. Katzenelson, W. McAllister, G. Rozas, and G.J. Sussman, "The Supercomputer Toolkit and its Applications," *Proc. of the Fifth Jerusalem Conference on Information Technology*, Oct. 1990.

[5] R. E. Bellman and S. E. Dreyfus, *Applied Dynamic Programming.* Princeton University Press, Princeton, New Jersey, 1962.

[6] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms.* The MIT Press, Cambridge, MA, 1990.

[7] R. C. Dorf, *Modern Control Systems.* Addison-Wesley, MA, 1989.

[8] J. Guckenheimer and P. Holmes, *Nonlinear Oscillations, Dynamical Systems, and Bifurcations of Vector Fields.* Springer-Verlag, New York, 1983.

[9] C. S. Hsu, "A Discrete Method of Optimal Control Based upon the Cell State Space Concept." *J. Optimization Theory and Applications,* 46(4):547-569, August 1985.

[10] A. Isidori, *Nonlinear Control: An Introduction.* Springer-Verlag, New York, 1985.

[11] F. C. Moon, *Chaotic Vibrations.* John Wiley & Sons, New York, 1987.

[12] F. C. Moon and P. J. Holmes, "A Magnetoelastic Strange Attractor." *J. Sound Vib.*, 65(2), 1979.

[13] Y. Sakawa and Y. Shindo, "Optimal Control of Container Cranes." *Automatica,* 18(3), 1982.

[14] J. E. Slotine and W. Li, *Applied Nonlinear Control.* Prentice-Hall, Englewood Cliffs, New Jersey, 1991.

[15] J. J. Stoker, *Nonlinear Vibrations.* Interscience, New York, 1950.

[16] F. Zhao, "Automatic Analysis and Synthesis of Controllers for Dynamical Systems Based on Phase Space Knowledge." *PhD Thesis Proposal*, Dept. of EECS, MIT, May 1, 1990.

[17] F. Zhao, "Extracting and Representing Qualitative Behaviors of Complex Systems in Phase Spaces." To appear in *Proc. of the 12th International Joint Conference on Artificial Intelligence*, Australia, August 1991. Also available as *MIT-AIM-1274*, MIT Artificial Intelligence Laboratory, March 1991.