Optimum Actuator Selection with a Genetic Algorithm

James L. Rogers
NASA Langley Research Center

University at Buffalo

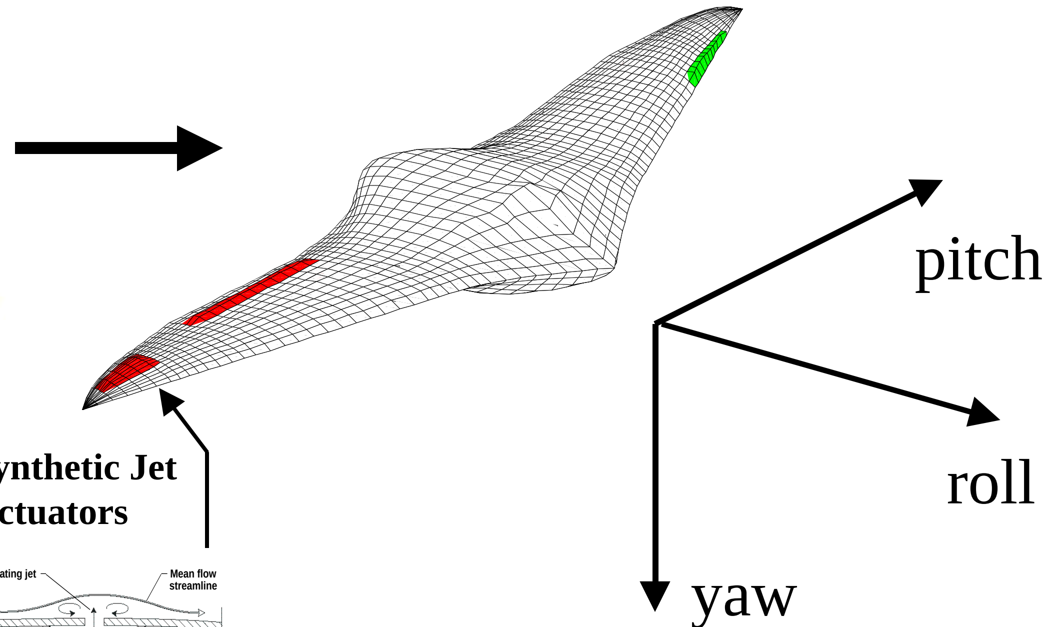
March 22, 2001

Outline

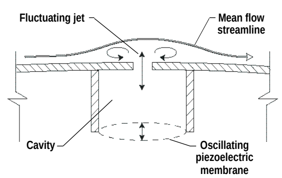
- Background and models
- Genetic Algorithm (GA) approach
- NACA airfoil model
    - Single processor results
    - Parallel processor results
    - Symmetry results
- Timing results
- Seamless aircraft model
- Concluding remarks

Why Do This Research?

Example of a seamless aircraft



Synthetic Jet Actuators



Replace conventional control devices like flaps and ailerons with synthetic jet actuators to create a seamless aircraft with no moving control surfaces

Problem Statement

Problem
Minimize the number of actuators needed to provide the uncoupled moments about the pitch, roll, and yaw axes.

Concern
Developing control laws is a time consuming process. Only the most promising configurations should be presented to the Controls specialist.

Our task
Develop software tools to significantly reduce the time required to optimally select and distribute the actuators over the aircraft surface.

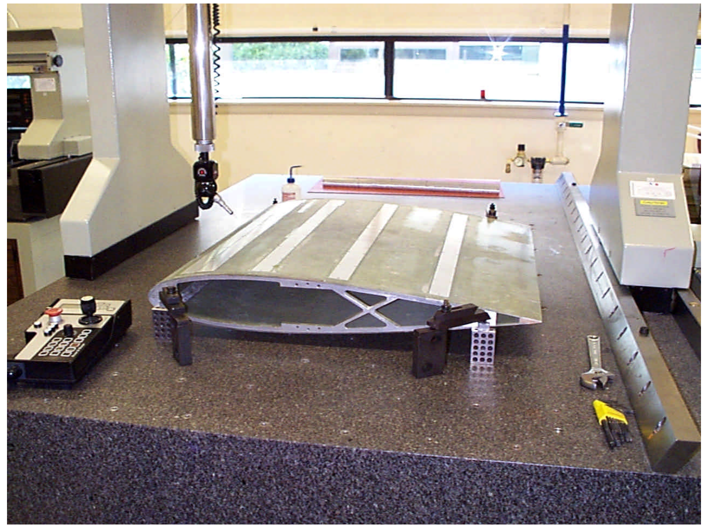
Phase 1 - Develop tools for a simplified model as a proof of concept
- Single processor
- Parallel processor

Phase 2 - Expand to a more complex seamless aircraft model

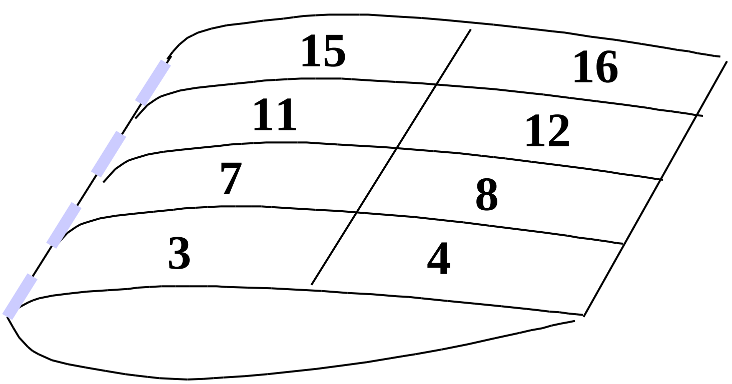
Simplified Model

Untapered, unswept wing based on NACA 0015 airfoil

Wind tunnel model

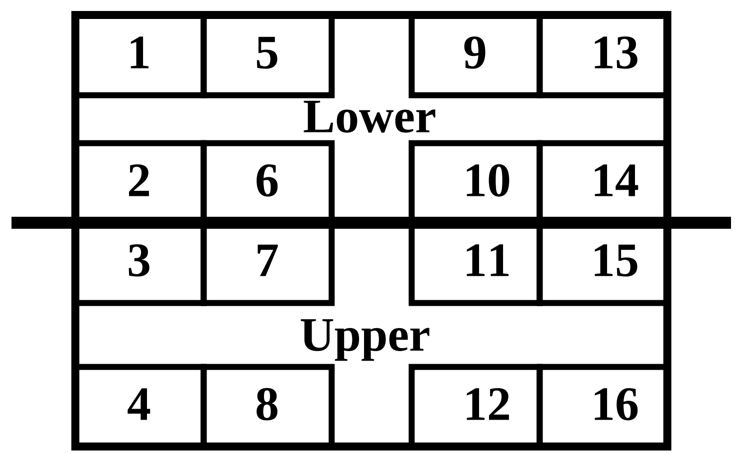


Analysis model

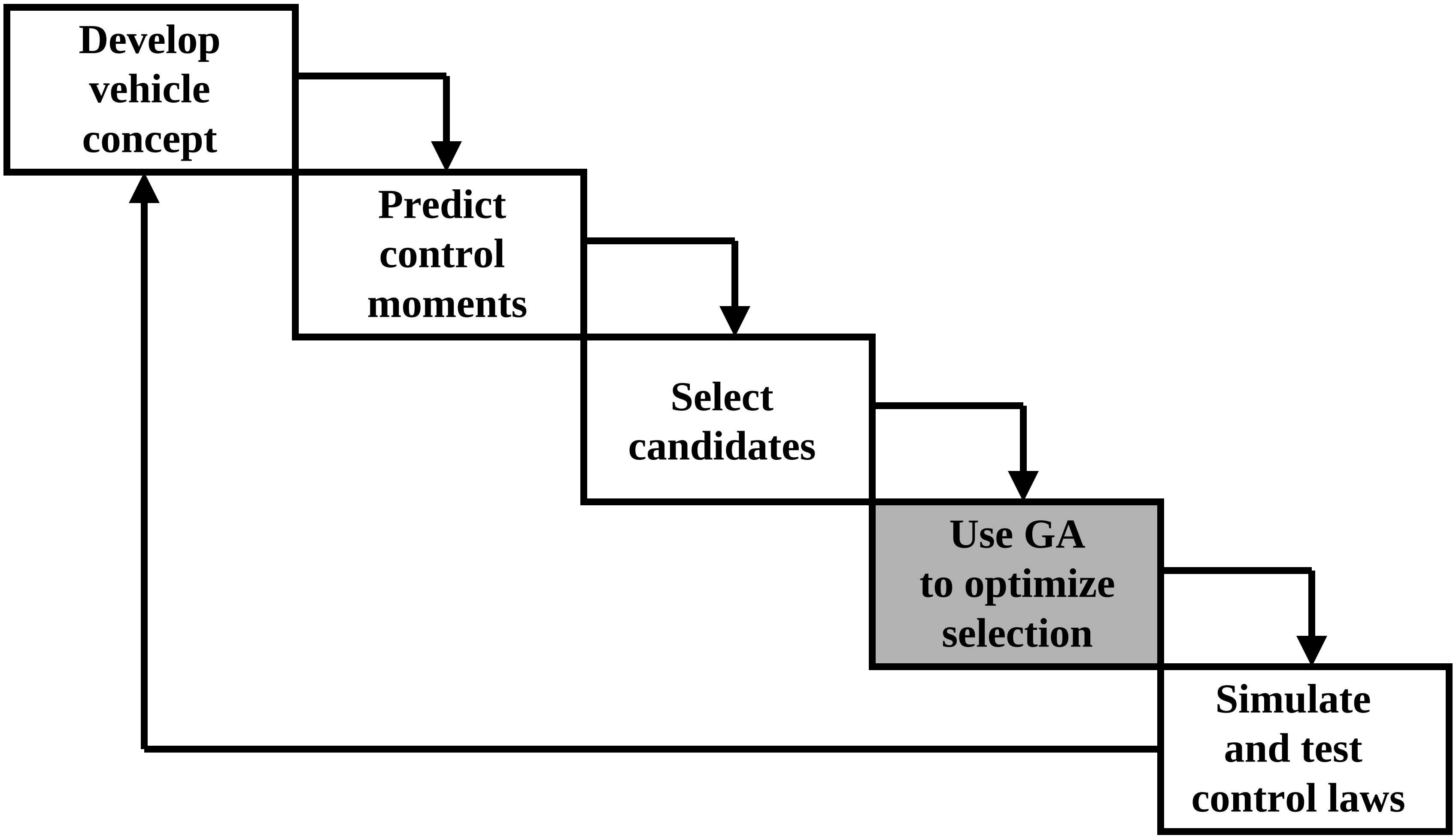


Leading Edge

Unwrapped model



Control System Design Process



Multi-Objective Application

(One Objective for Each of Pitch, Roll, and Yaw Subproblems)

Given 16 actuator locations, find the minimum number of actuators and their placement to provide uncoupled pitch, roll, and yaw moments.

Penalize the objective function for the **pitch subproblem** if:

- $|C_l| > .001$
- $|C_n| > .001$
- number actuators < 2 (take advantage of engineering knowledge)
- $|\mathbf{C_m}| < .001$

Similar penalties for the roll and yaw subproblems.

Genetic Algorithm Approach

- Rapidly examine a large number of candidate actuator placements.
- Select the optimum placement based on the minimum number of actuators as well as the moment and coupling data.
- The fitness of a population member is determined by calling a 3D, low-order, potential-flow panel program. Must have very fast function evaluations because it is called so often.
- Penalize fitness if constraints are violated.

GA Information

- Population size = 100 (different populations for each subproblem)

- Population member - string of length 16 (0 1 0 1 0 0 1 1 0 1 1 1 0 0 0 1)
  1 indicates an active actuator while 0 indicates an unused actuator

- Fitness function = sum of active actuators plus constraint (if any)

- Absolute values used for moments

GA Operations

Selection - based on fitness

Tournament approach retains the best patterns for next generation

0011001100110011 f(x) = 8 → Tournament → 1000000100000010

1000000100000010 f(x) = 3 → Tournament

Single point crossover - combines features of two parents

Parent 1 - 1 1 1 1 1 1 Parent 2 - 0 0 0 0 0 0

Randomly generated crossover point - 2

Child 1 - 1 1 0 0 0 0 Child 2 - 0 0 1 1 1 1

Mutation - introduces new patterns, rate = .01

Before - 0 0 0 0 0 0

Randomly generated mutation point - 4

After - 0 0 0 1 0 0

Computing the Composite Fitness

Multilevel Optimization

The string 0 1 1 0 0 0 1 0 0 0 0 1 0 0 1 0
indicates there are actuators in locations 2 3 7 12 and 15

Composite fitness computed using an OR function

Location	1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6
Pitch (4)	0 0 0 1 0 1 0 0 0 1 0 0 0 0 0 1
Roll (4)	0 0 0 0 0 0 1 1 1 1 0 0 0 0 0 0
Yaw (4)	0 0 0 0 0 0 0 0 0 1 1 0 1 0 0 1
Composite (9)	0 0 0 1 0 1 1 1 1 1 1 0 1 0 0 1
Location	1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6

Problems Encountered and Resolved

- Crossovers kept producing the same strings
  Corrected by only crossing different strings

- Originally looked at composite strings inefficiently by computing the composite a member at a time, for example:

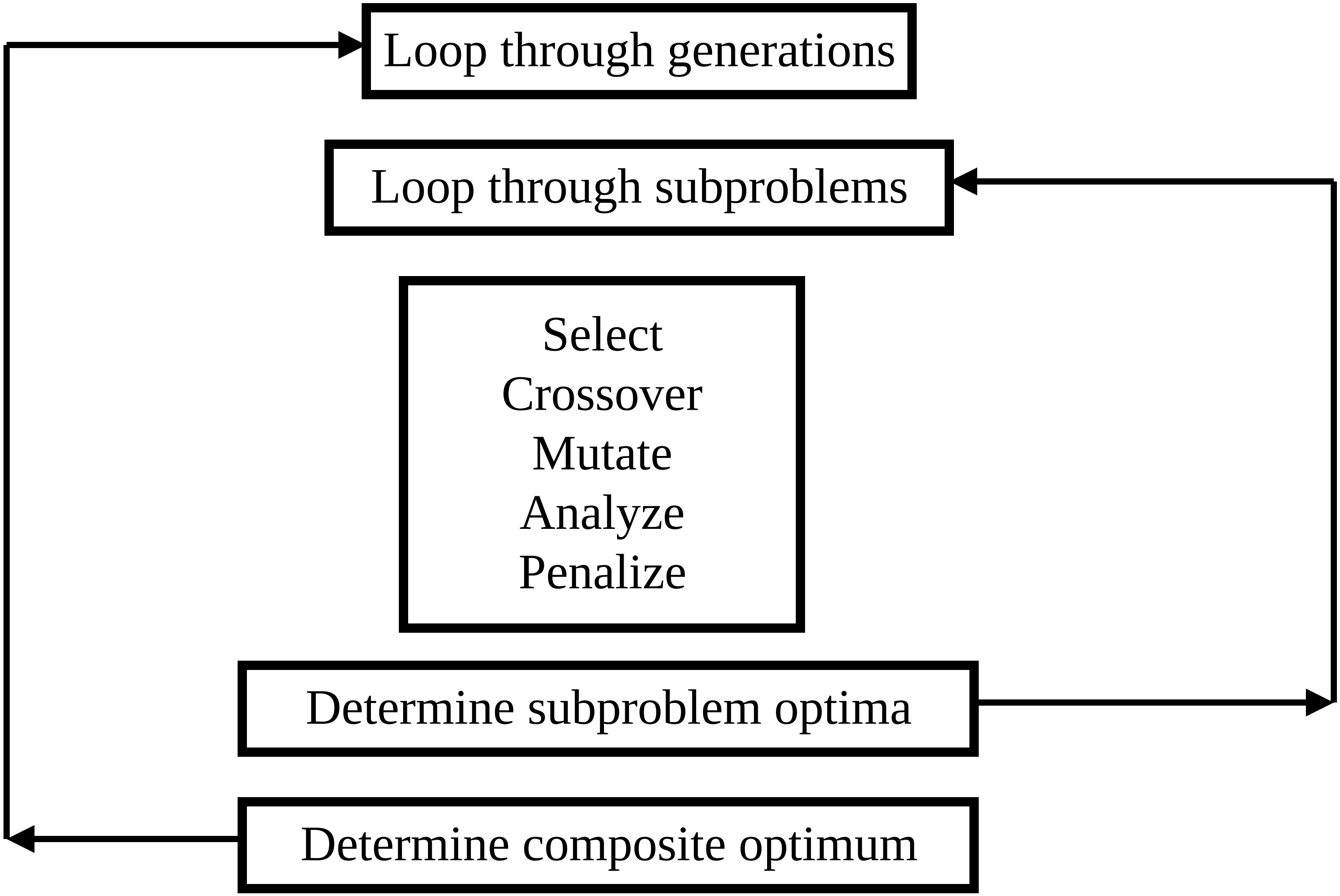
Member 5	pitch = 4	roll = 10	yaw = 4	composite = 13
Member 10	pitch = 10	roll = 4	yaw = 10	composite = 10

Corrected by saving all valid strings and comparing

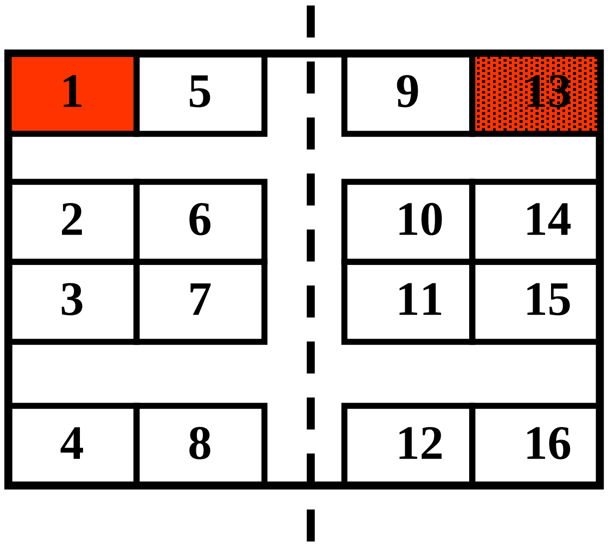
pitch = 4 roll = 4 yaw = 4 composite = 9

Single Processor Flow

Multi-objective and Multi-level

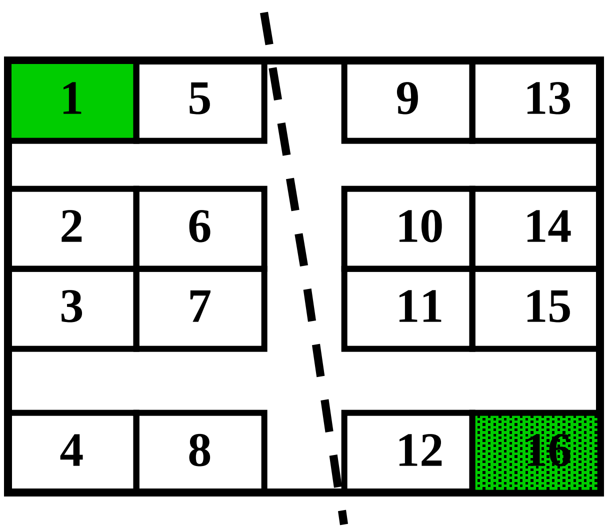


Wing Symmetry

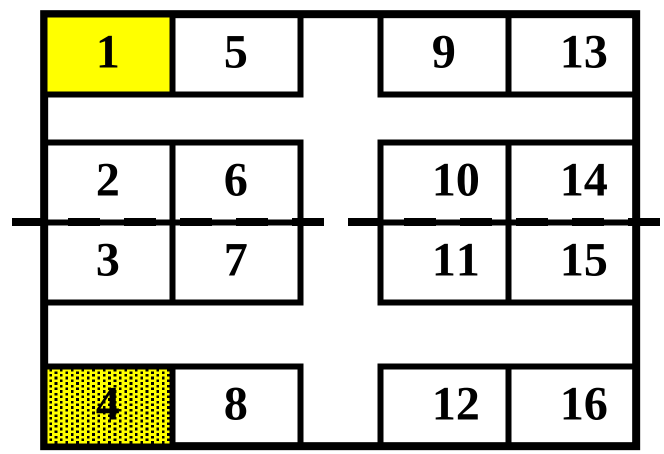


Pitch symmetry left to right

Wing model is symmetric so information can be used to determine a composite model for all six uncoupled moments



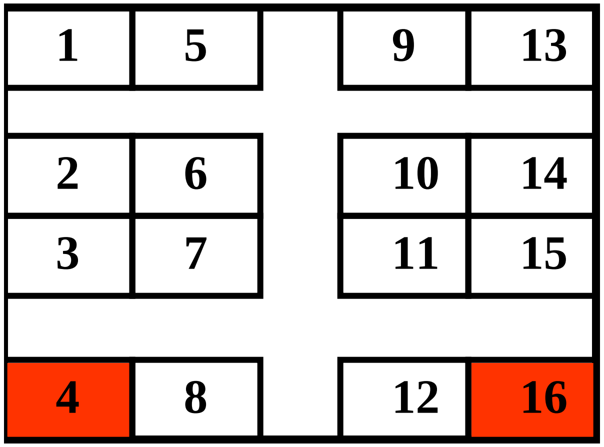
Roll symmetry top left to bottom right



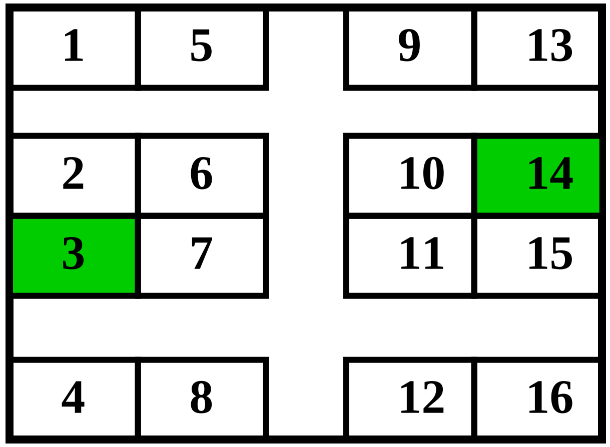
Yaw symmetry top to bottom

Actuator Placement
(Single Processor - 65 hours)

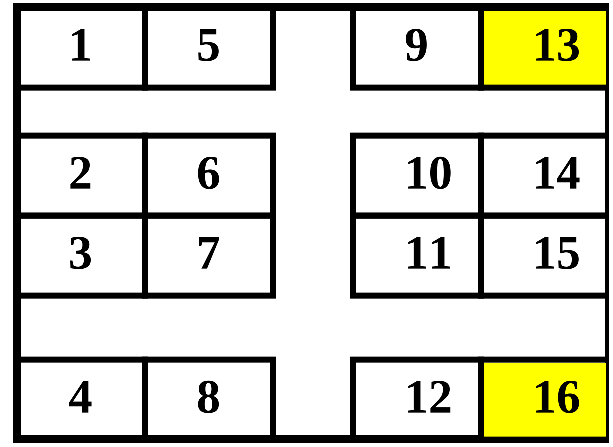
Pitch up



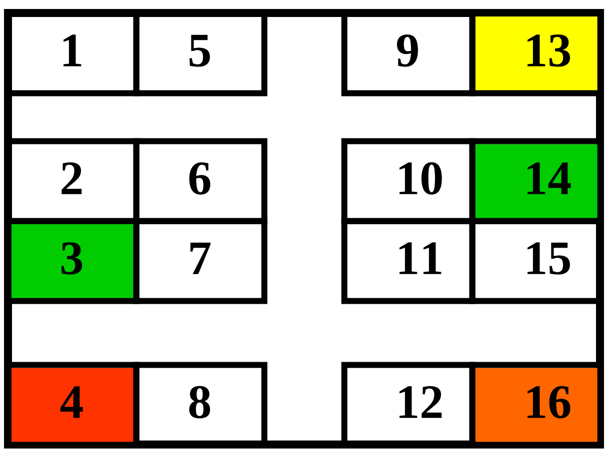
Roll right



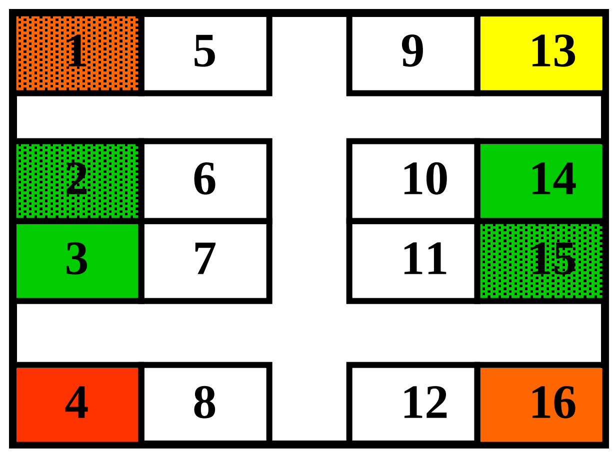
Yaw right



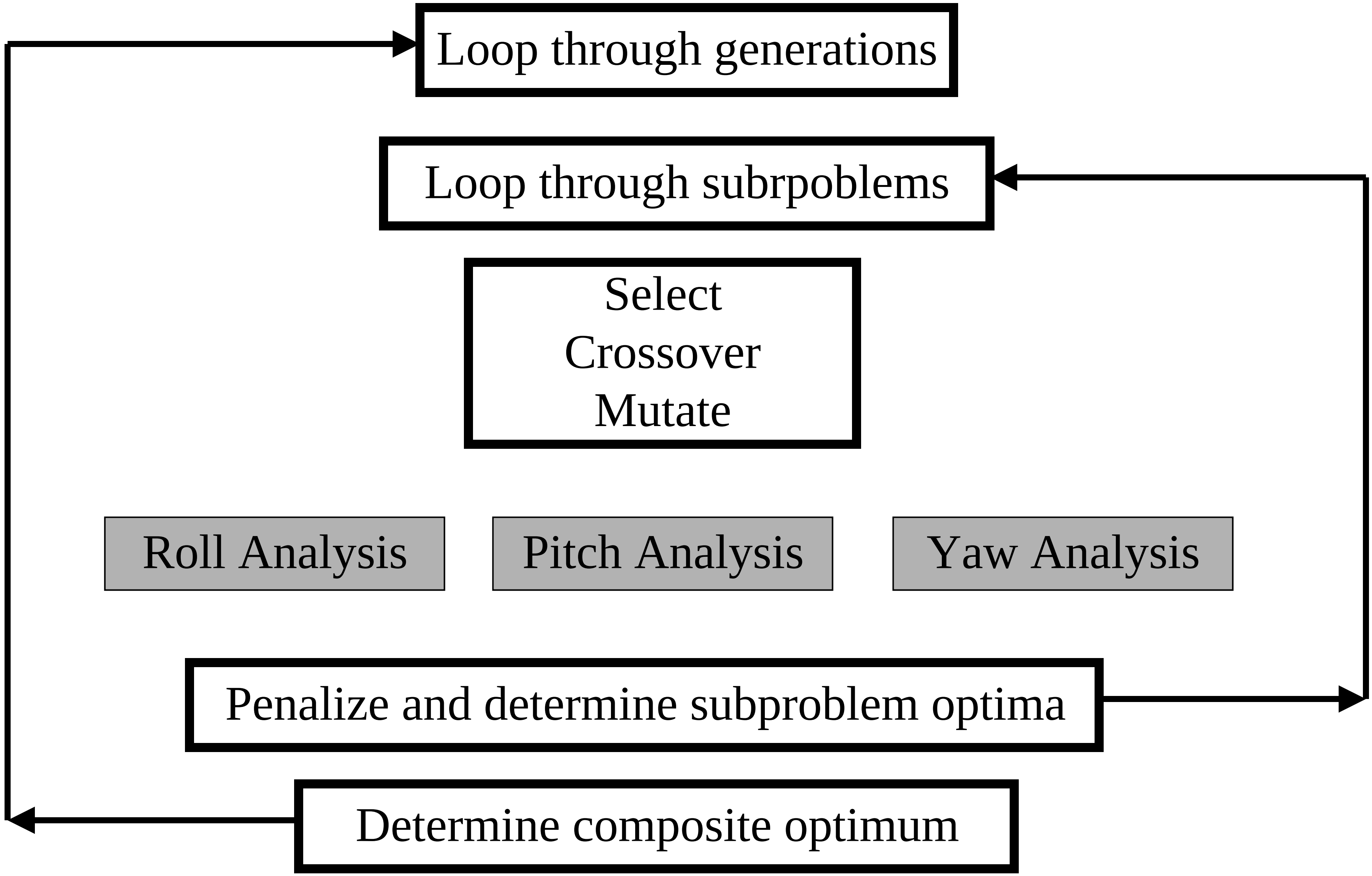
Three maneuvers



Six maneuvers

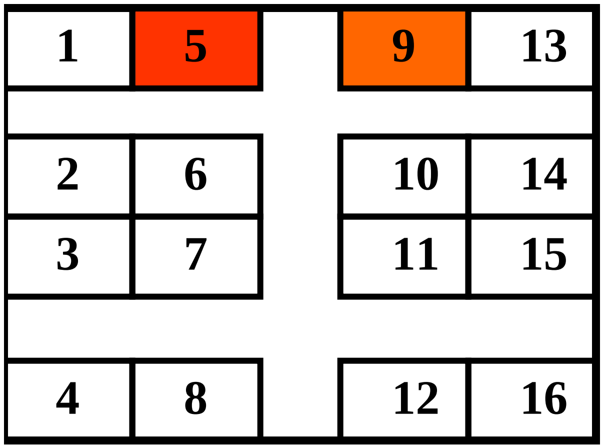


Parallel Processor Flow

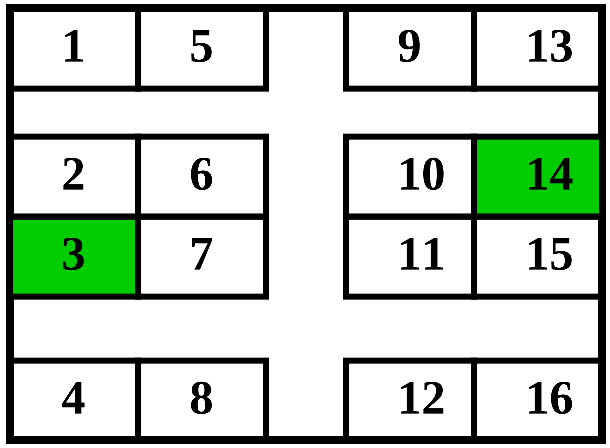


Actuator Placement
(Parallel Processors - 22 hours)

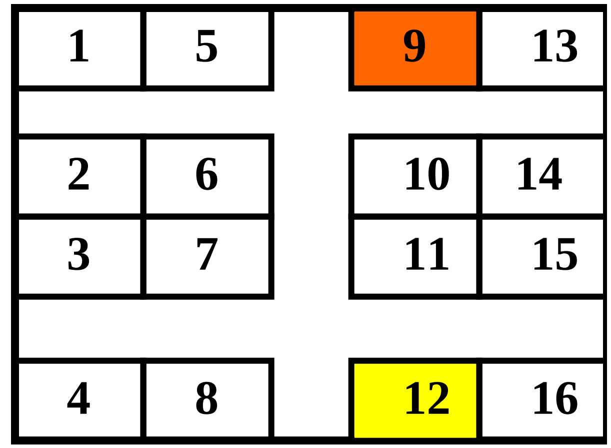
Pitch down



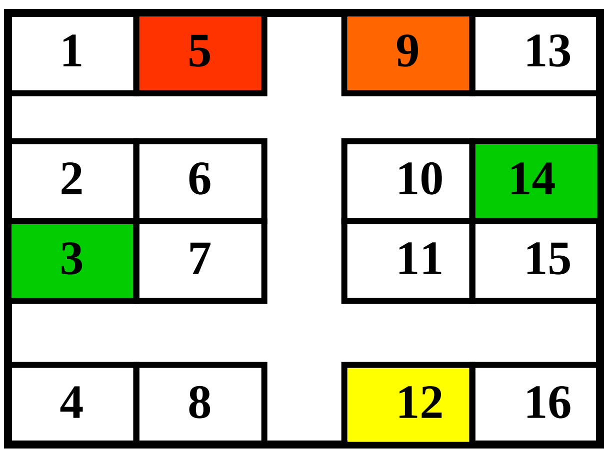
Roll right



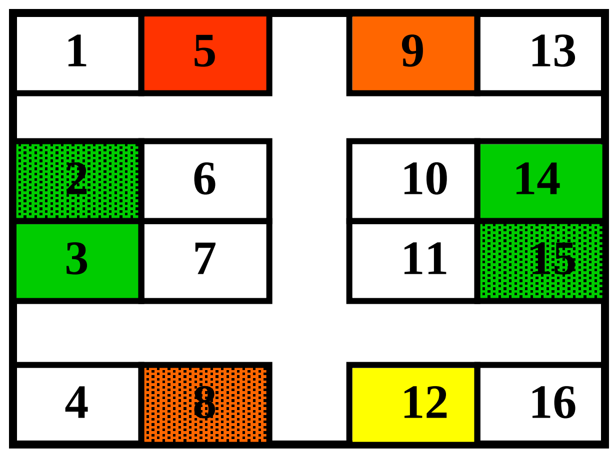
Yaw right



Three maneuvers



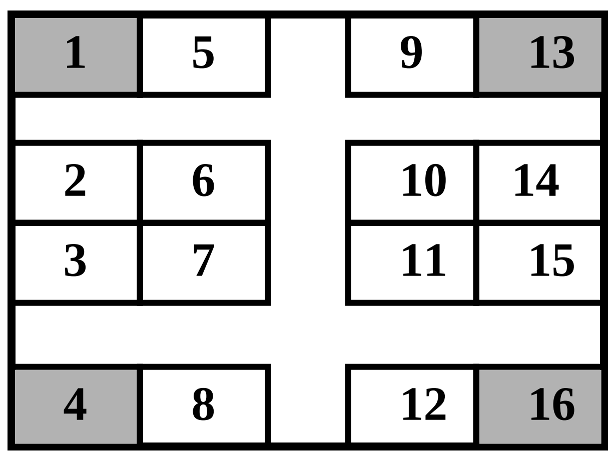
Six maneuvers



Symmetry Enhancement

- Refined GA approach to take more advantage of wing symmetry
  - Reduces design space by using only 8 locations
  - 256 possible combinations
  - Reduces member length and population size
  - Finds optimum in one hour (20 minutes if done in parallel, estimated)

Six maneuvers

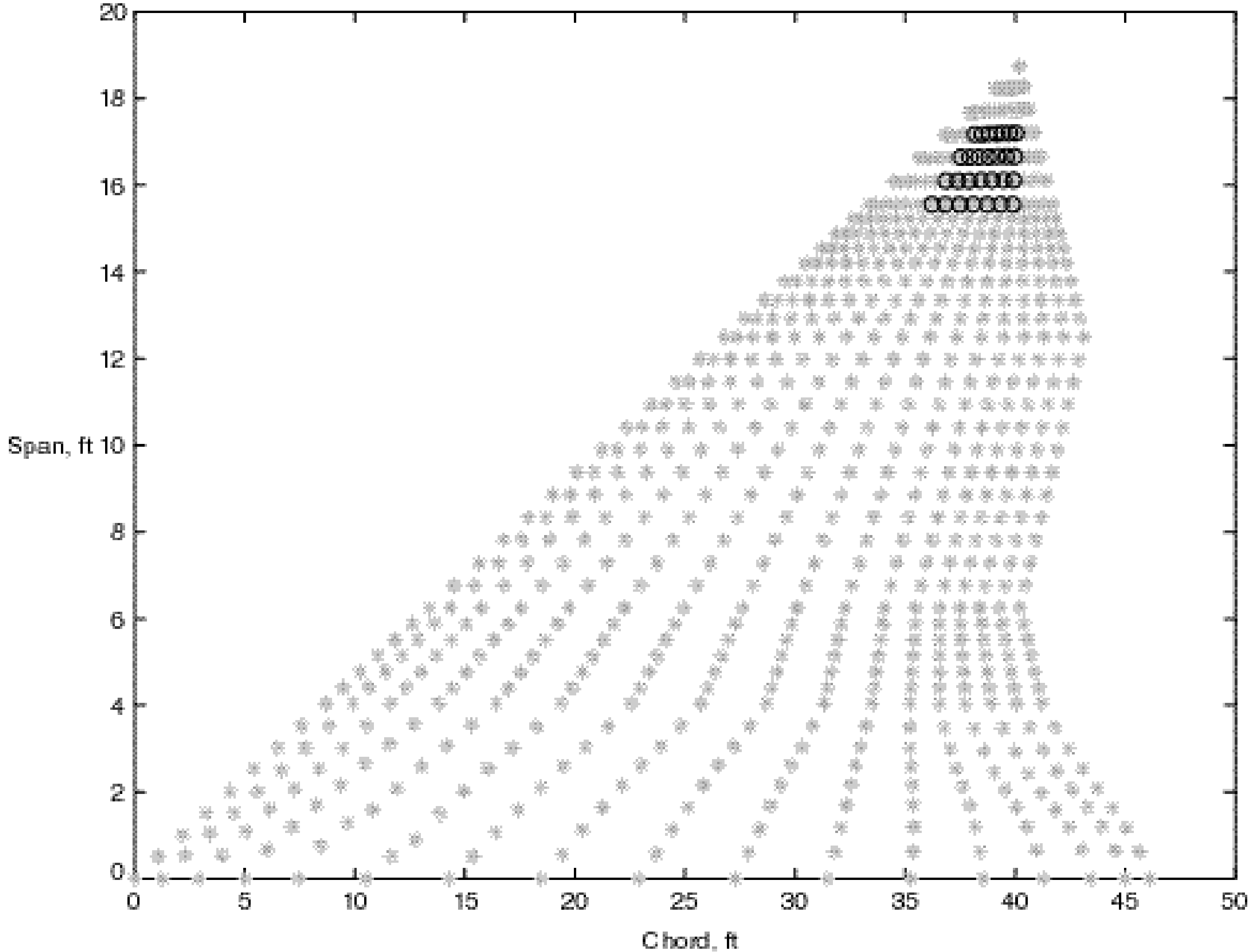


Timing Results

Each analysis takes one minute.
GA has 300 analyses per generation with 13 generations (3900 analyses).

Actuators	Combinations	Search Design Space	GA Time
16	65,536	~1,100 hours	65 hours (one processor)
16	65,536	~1,100 hours	22 hours (multi-processor)
16	256	~ 4 hours	1 hour (symmetry)
34	1.7B	~ 286M hours	
100	Do not even think about it!		

Seamless Aircraft Model

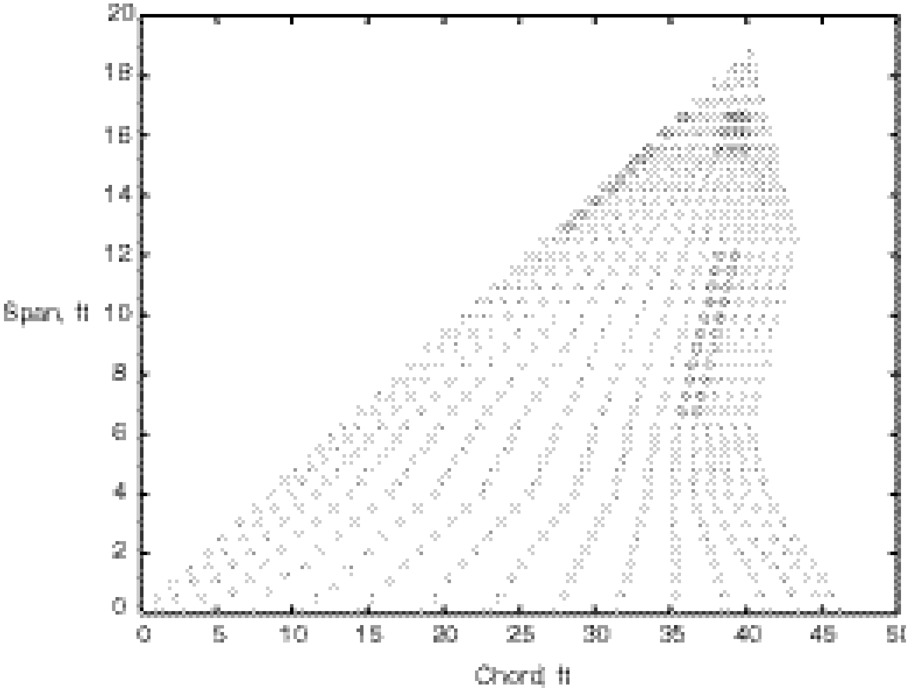


One of 34 candidate effector arrays

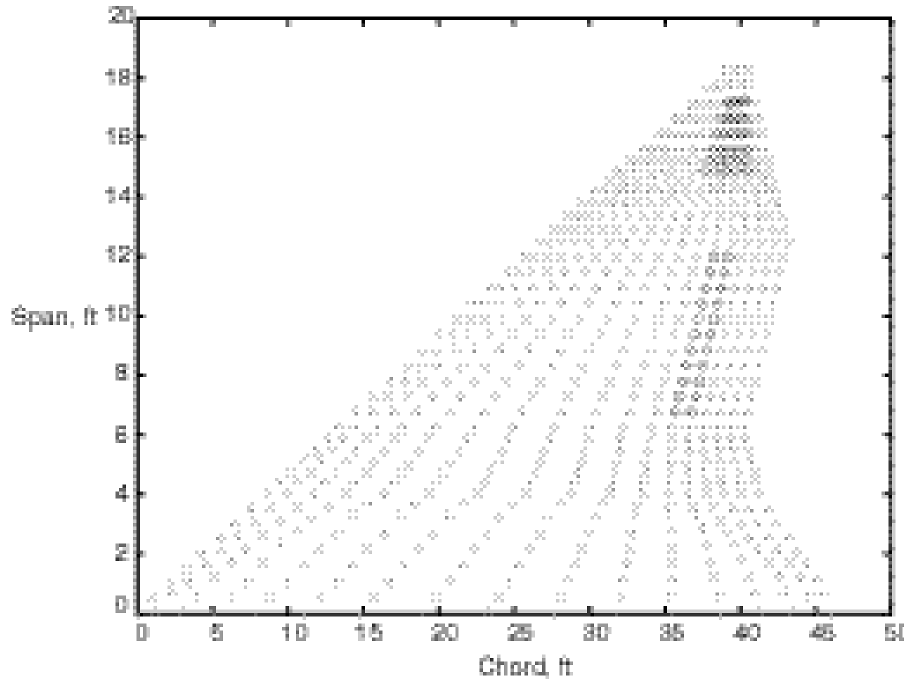
Project Set Up

- Seven possible locations for effector arrays
  - Upper wing - trailing edge, leading edge, tip, and mid chord
  - Lower wing - trailing edge, leading edge, and tip
  - Each location has eight options (including no array)
  - Can select at most one option from each location
  - Possible combinations - ~ 5 million
- MATLAB used to simulate analysis for fitness function
  - Penalty function used
- Population size - 200
- 300 generations requires about 1 hour of time
  - Evaluated about 60,000 combinations

Results



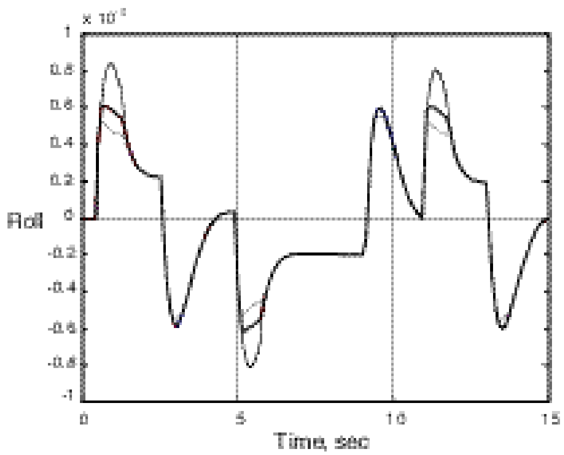
Upper Surface



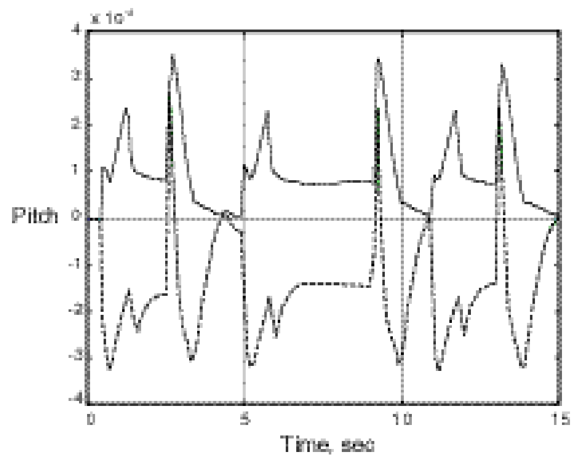
Lower Surface

- GA selects five arrays with 96 devices which met all requirements
- Engineer had manually chosen four arrays with 82 devices, but did not meet all requirements when tested

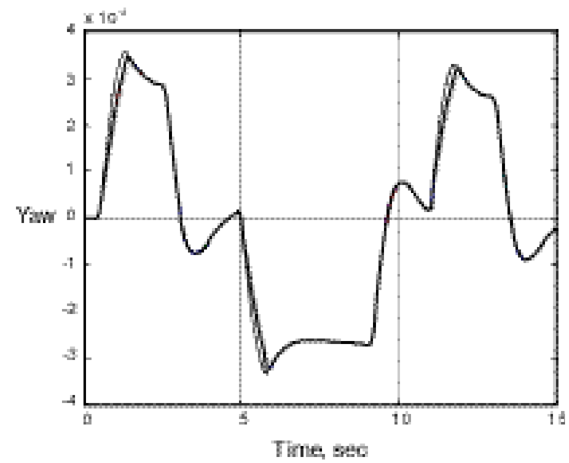
Test of Simulated Controller



Roll



Pitch



Yaw

GA — Ideal — Manual

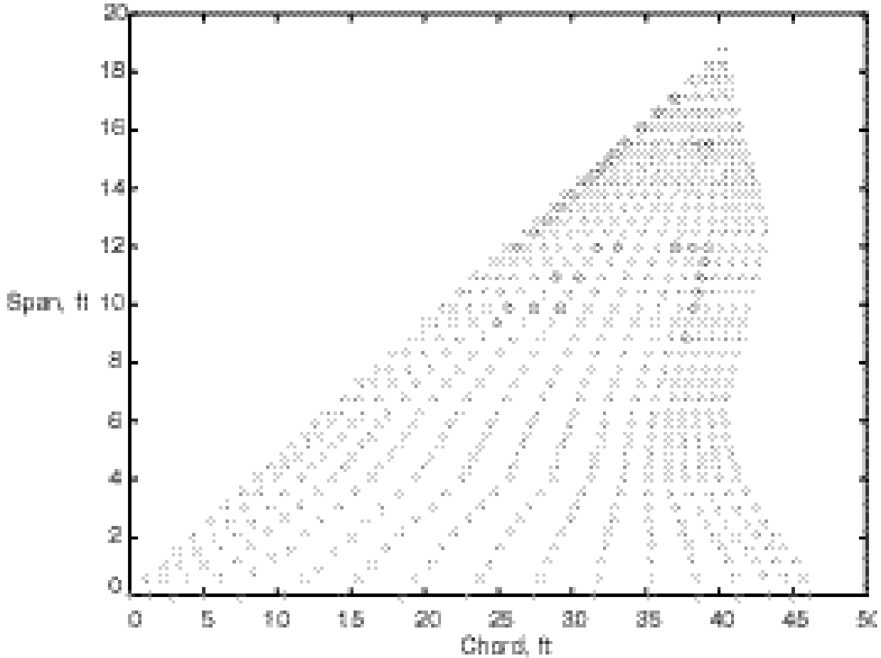
GA arrays perform better than manually selected arrays for roll and yaw

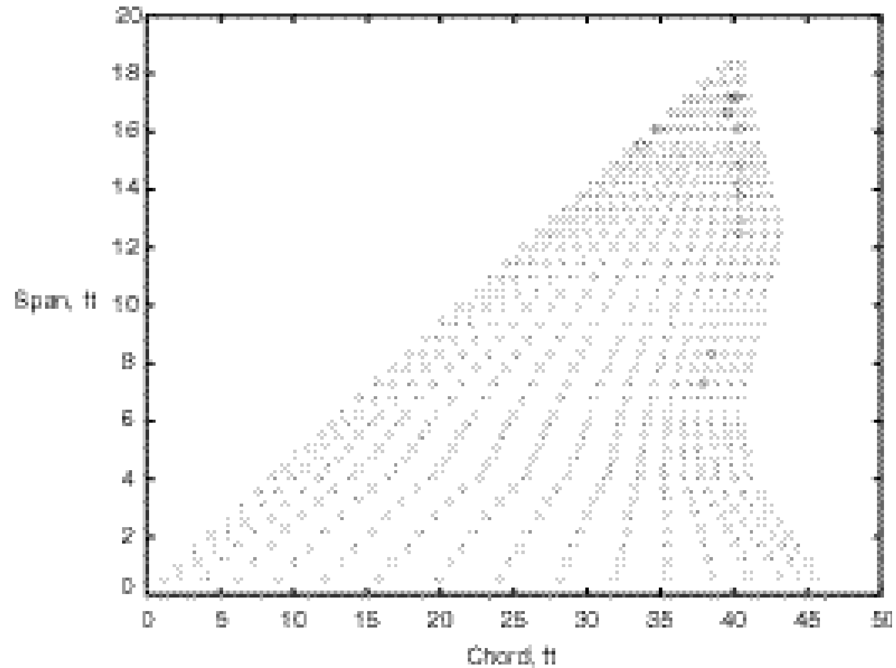
Both sets of arrays cause an undesired pitch perturbation, but the GA results in smaller pitch transients during the maneuver

Individual Devices

- 349 possible locations
- String length of 100
- Possible combinations - $4x10^{99}$
- Population size 300 and 500 generations
- No duplicates allowed in the string
- 8 hours run time

Results





Upper Surface

Lower Surface

Only 45 devices needed to provide control

No simulations were done with this model.

Concluding Remarks

Research is
Seeing what everyone else sees but thinking what no one else has thought!

A problem that once appeared to be unsolvable using enumeration, now looks promising with the application of a genetic algorithm.