Complete Guide for Building UAV

David Hyunchul Shim

University of California, Berkeley

IEEE CCA UAV Workshop

September 24, 2000

hcshim@robotics.eecs.berkeley.edu
http://robotics.eecs.berkeley.edu/~hcshim

Overview

- Overview of UAV system
- Selection of vehicle platform
- Sensor system
- Model Identification
- Hierarchical Control System
  - Low-level vehicle stabilization and control
  - Way-point navigation
- Further applications

Overview of UAV system

Why UAV?

- Needs for remote and autonomous operation

  Surveillance, remote photography

- Testbed for advanced control laws

Why Rotorcraft-based UAV?

- Versatile flight modes
- Relatively small space is required to operate (no runway)

Downsides: inherently unstable: difficult to control manually

complicated MIMO nonlinear dynamics

complex mechanism (more expensive, more maintenance)

UAV Design Procedure

Objective Definition
- -What do we need UAV?
- -What do we want to do with UAV?

Vehicle Selection
- - What functions do we want from UAV?
- - What size of UAV do we need?

Avionics Selection
- -What sensor do we need?
- -What kind of computer is good for us?

Control System Implementation
- -Helicopter model identification
- -Controller Design (classical/modern control)
- -Building hardware, mounting, enclosure…

Flight Test
- -Careful planning is mandatory for safety

UC Berkeley BEAR TEAM AIAA Aug. 14, 2000

Key points of Successful RUAV Implementation

- **Payload problem**

  Find a helicopter powerful enough to carry necessary sensors&computers
- **Navigation sensor integration**

  Implement accurate INS/GPS combinatory sensor
- **Helicopter Control Model Identification**

  Obtain high-fidelity control model to design stabilizing low-level controller
- **Hardware/software integration**

UAV base platform selection

Selection Points

- PAYLOAD is the most dominant factor for now.

  usually depends on the size of the helicopter

  at least 4kg of payload is required (60 class glow engine is marginal)
- There are limited choices for base RUAV platform
- Integration is difficult because candidate helicopters were not designed for UAV application

UAV base platform selection

Integration

- Mechanical aspect

  Vibration isolation is the primary issue.

  Sensor mounting: IMU needs careful mounting for proper vibration isolation

  Computer mounting: shock vibration for prolonged lifetime and reliability

- Electronic aspect

  Need to interface with servomotors (usually accepts PWM signal)

Specifications of Berkeley RUAVs

Name	Length	Height	Width	Weight Payload	Engine	Autonomy
Kyosho Concept 60	1.4 m	0.47 m	0.39 m	4.5kg airframe 5 kg payload 4.8kg avionics	OS FX91 Glow engine 2.8bhp	Boeing DQI NovAtel RT-2 MediaGX233
Bergen Industrial Twin	1.5 m	0.7 m	0.3 m	7kg dry weight 10 kg payload	Twin Genoa Gas Engine	N/A
Yamaha R-50	3.58 m	1.08 m	0.7 m	44kg airframe 20 kg payload 10kg avionics	Water-cooled 2 stroke 1 cylinder gas engine 98cc, 12ps	Boeing DQI NovAtel RT-2 Pentium233 2 ultrasonic altimeter Vision processor
Yamaha RMAX	3.63 m	1.08 m	0.72 m	58 kg dry weight 30 kg payload	Water cooled 2 stroke 2 cylinder Gas engine 256cc, 21ps	Boeing DQI NovAtel RT-2 Dual Flight Computer Dynamic Network Router Vision processor Digital compass

Berkeley BEAR Fleet: *Ursa Minor3 (1999-)*

Length: 1.4m
Width: 0.39m
Height: 0.47m
Weight: 9.4 kg
Engine Output: 2.8 bhp
Rotor Diameter: 1.5m
Flight time: 15 min
System operation time: 30 min

Bergen with shock-absorbing landing gear

- Pneumatic-operating shock-absorbing landing gear

Length: 1.5m
Width:0.3m
Height: 0.7m
Dry Weight: 8 kg
Payload: 10kg

Berkeley BEAR Fleet: *Ursa Magna2 (1999-)*

Based on Yamaha R-50 industrial helicopter

Length: 3.5m Width:0.7m
Height: 1.08m
Dry Weight: 44 kg Payload: 20kg
Engine Output: 12 hp
Rotor Diameter: 3.070m
Flight time: 60 min
System operation time: 60 min

UC Berkeley BEAR TEAM — AIAA Aug. 14, 2000

Berkeley BEAR Fleet: *Ursa Maxima 1 (2000-)*

Based on Yamaha RMAX industrial helicopter

Length: 3.63m Width:0.72m
Height: 1.08m
Dry Weight: 58 kg Payload: 30kg
Engine Output: 21 hp
Rotor Diameter: 3.115m
Flight & system operation time: 60 min

UC Berkeley BEAR TEAM — AIAA Aug. 14, 2000

Onboard Flight Control System

Tasks of onboard computer system

- Sensor Management
  INS/GPS integration, ultrasonic sensor polling, etc
- Control output computation
  Stabilizing control law computation and PWM signal generation
- Communication
  Wireless communication among ground station, DGPS base station, other UAVs
  RS-232 serial communication onboard vision computer

Choice of Navigation sensors

Guideline

- **Velocity, attitude angle and angular rates(optional) are necessary to stabilize the inherently unstable RUAV.**
- **Position estimate is required to guide the RUAV through waypoints.**

Other considerations

- Mechanical properties
  size, weight, vibration characteristics, mounting points
- Electronic properties
  supplied power (DC 12V is preferred)
  interface method (RS-232, analog output)
- Cost!!!

Choice of Navigation sensors

Types of Navigation Sensors

- Inertial Navigation System (INS)
- Global Positioning System (GPS)
- Vision system
- Ultrasonic sensor
- Laser range finder
- Radar
- Air pressure/Barometric sensor

INS/GPS are usually favored as the primary navigational sensor system due to their high accuracy and complementary nature

Choice of Navigation sensors

- **Inertial Navigational Sensor**

  three accelerometers and three rate gyros installed orthogonally
  accelerometers: typically solid-state micromachining technology
  rate gyro: solid-state or fiber-optic gyro

Characteristics

Self-contained, high update rate
Numerical integration, bias estimation, alignment are required
By itself, estimation will diverge sooner or later (depending on the sensor quality) ➡ external correction is required

Choice of Navigation sensors

- **IMU or INS?**

IMU (Inertial Measurement Unit)
Accurate timing can be maintained
INS performance can be tailored to suit the application
Usually lighter than INS system

Sophisticated integration, bias estimation, alignment should be performed.
Numerical process may be demanding (relatively low sampling)
ex) *Systron-Donner MotionPak, Litton LN-200*

Choice of Navigation sensors

INS
Indicates a system sensor and processor integrated in one package
Built-in DSP chips can process the sampled data at very high rate
GPS integration can be readily achieved.
Complete navigation solution can be read by serial port.
Can exploit the advanced inertial navigation technology.
Numerical burden on FCS can be avoided.

Tuning the sensor can be difficult
Integration of GPS has limited option
Maintaining correct timing can be difficult.

ex) *Boeing DQI-NP, Litton LN-250 Guidance Package (GGP)*

Choice of Navigation sensors

So, IMU or INS?

Enough process power available from the main FCS?
Advanced integration, bias estimate, alignment algorithm accessible?
Need to tune the INS in a special way to suit the application?
Need a special way to integrate GPS?
Cost?

Navigation Hardware: INS

INS: Boeing DQI-NP

- 3 Accelerometers, 3 rate gyros sampled at 600Hz
- Inertial measurements processed by DSP
- Position, linear velocity, Euler angle, angular rates
- Computation output rate at 1, 10, 100Hz via RS-232 (115200bps)
- Accepts GPS measurement via RS-232 (9600bps) compatible with PLGR and MicroTracker only
- Used data set: *M3512@100Hz M3@1Hz*
- *Excessive amount of serial communication load due to M3512@100Hz*

Choice of Navigation sensors

- **Global Positioning System**

Estimates the current position relative to ECEF coordinate system using radio signals coming from 24 non-geosynchronous satellites 20,000 km away from the surface of the earth.
Accuracy: 2cm~2.5m depending on algorithm

- Error is bounded regardless of time
- Update is relatively slower than INS (up to 10 Hz)

- Takes relatively longer time to lock up than INS initialization
- The satellite lock-up can be fragile and varies with the time of the day.
- Excessive roll/pitch angle can cause poor signal reception
- The card may be sensitive to mechanical vibration

Navigation Hardware: GPS

GPS: NovAtel MillenRT-2

- DGPS corrected position measurement available up to 10Hz
- Many different logs available
- Measurement available via RS-232 (115200bps)
- Measurements are used to compute position at higher rate (50Hz) and relayed to Boeing DQI-NP
- Log data: *PRTKB@4Hz, PXYB@1Hz*

Choice of Navigation sensors

- **Vision-based navigation**

  Relative sensor: estimates navigation information using image
  Relatively new approach- few successes have been reported (CMU)
  Heavy numerical load: custom DSP system is usually required

  UC Berkeley: color tracking and special feature board approach have been successfully tested

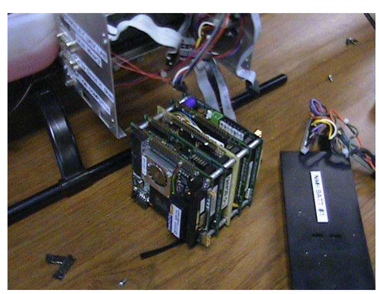
- **Ultrasonic sensor**

  Relative sensor: measures distance from helicopter to the ground
  required for autonomous take-off/landing
  Simple, easy-to-use, low-cost
  May not work properly on porous, uneven surface
  Extremely sensitive to mechanical vibration
  Special filtering algorithm should be used to correct the spiky reading

Navigation Hardware: Flight computer

Tasks

- Sensor management (RS-232, A/D converter, counter)
- Control output calculation
- Control output generation (PWM)
- Communication



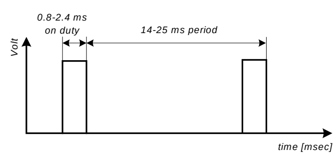
Main board requirements

- Reliability (robust to vibration- PC104)
- Multiple serial ports
- PWM generation and reading
- Wireless communication via RS-232 or Ethernet
- DC/DC power conversion board to run on DC power

Navigation Hardware: PWM Signal

Servo control: PWM signal

- Commands the servomotor output to certain angles- controls helicopter main rotor collective pitch, longitudinal/lateral cyclic pitch, tail rotor collective pitch and throttle angle of engine
- Output shaft is proportional to the on-duty interval



- PWM signal can be generated and read by 8254 counter/timer chip.
- PWM reading is very useful for system identification

Control Hardware: RX, Servo, TOB

Radio Transmitter/Receiver

- 72MHz PCM 9 channel radio receiver
- Generates PWM signals at 70 Hz

Servomotors

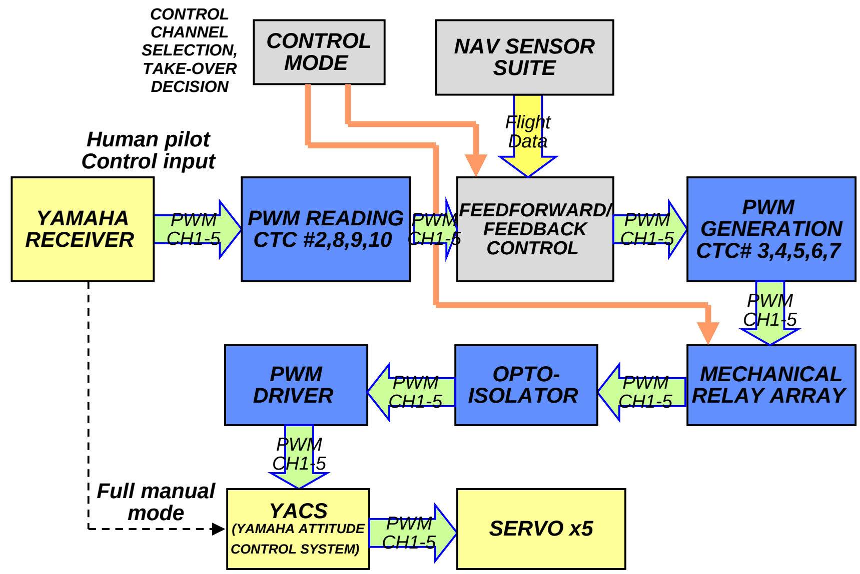
Accepts PWM at 40-70Hz

- Actuates servo proportional to the duration of "on-duty"

Take-over board with Counter/Timer Card (8254)

- Custom-made board with PC104 standard
- Reads servo control command from RX
- Generates servo control command at 50Hz
- Switches the control output between manual control and computer-generated control

Flight Control System signal flow



Navigation Hardware: Flight computer

Communication

- **Wireless modem**
  connected via RS-232 port
  limited bandwidth (up to 100 Kbps)
  longer range
  ex) *Freewave DGR-115 (140 MHz, 900 MHz, 2.4 GHz)*

- **Wireless LAN**
  High-bandwidth (up to 11 Mbps)
  Shorter range than wireless modem
  Versatile protocol (TCP/UDP/IP, telnet, ftp,…)
  ex) *Lucent WaveLAN, Cisco Aironet…*

Navigation Hardware: *Ursa Magna 2*

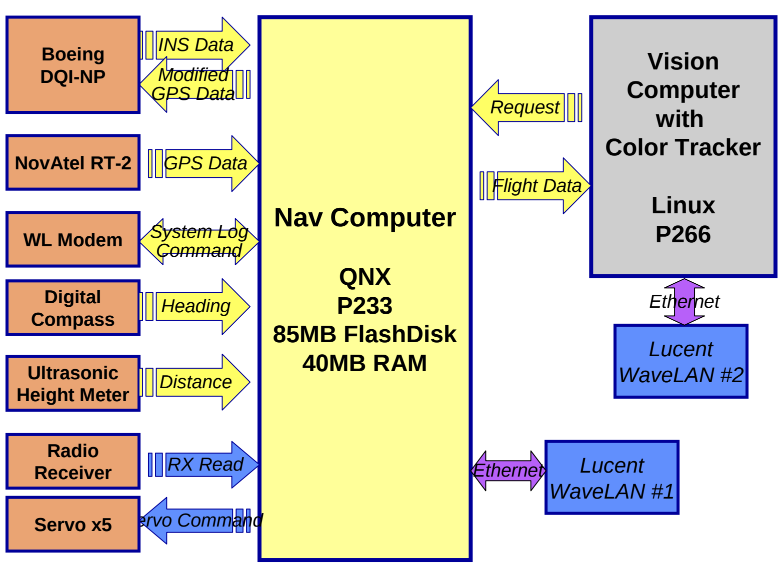
Flight computer: PC104 Littleboard P233

- Boeing DQI-NP INS
- NovAtel GPS MillenRT-2
- 2 Ultrasonic Sensors, 4 Contact Switches (take-off/landing)
- Lucent WaveLAN EC/S
- 2 Wireless modems: for long-range missions

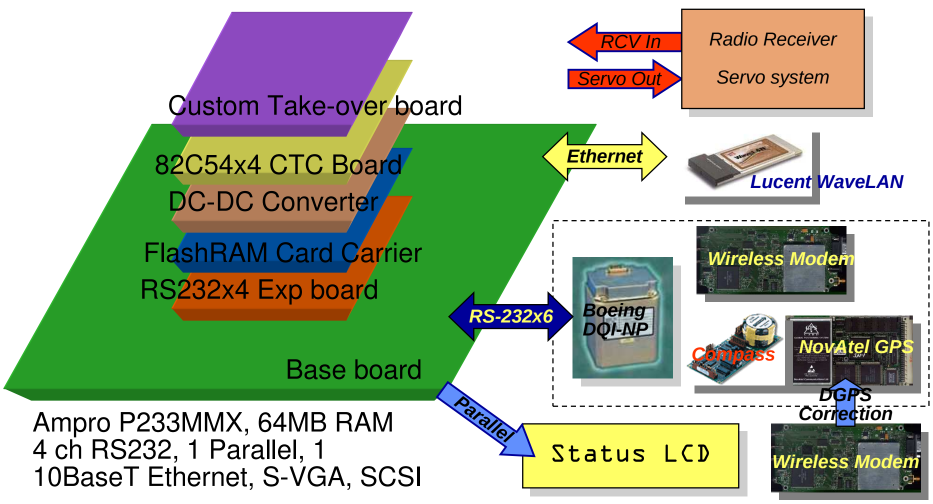
Vision Computer: PC104 Littleboard P266

- Sony Pan/tilt/zoom camera
- Frame grabber

Navigation Hardware: Onboard computer

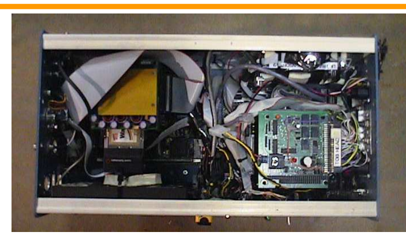


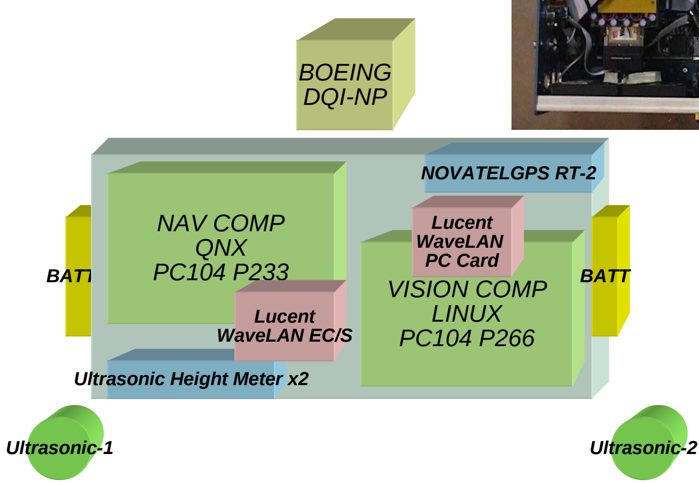
Navigation Hardware



Navigation Hardware: *Ursa Magna 2*

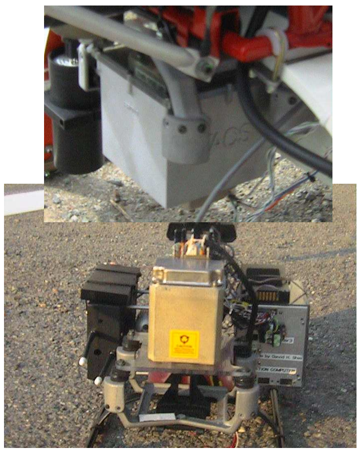
- Housed in 17x9x5.25" enclosure

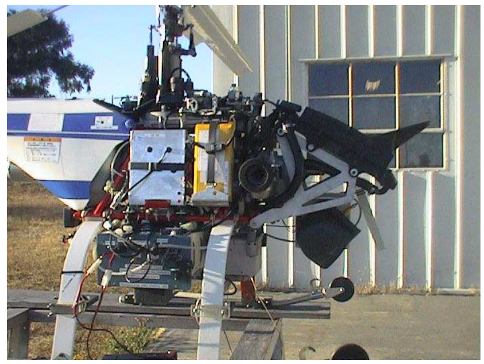




Mounting

- Proper shock isolation is necessary to reduce circuit failure
- INS needs special protection from engine/rotor vibration





Navigation Software: real-time OS

The OS should be able to handle

- Hard realtime requirements (50Hz sampling for control)
  low jittering of timing
- Multitasking (to support multi-rate process)
- Multiple serial port support
- Real-time Performance monitoring
- Convenient debugging
- Network (Ethernet)
- Portability, small size kernel, low memory requirement

Navigation Software: real-time OS

Choices

- MS-DOS
- MS-Windows
- RT-Linux
- VxWorks
- QNX

Navigation Software: real-time OS

Real-Time OS: QNX

- **Advantages**

- Systematic way to write real-time programs
- UNIX-like system architecture (POSIX4 compliant)
- Very compact kernel
- Inter-Process Communication (IPC) support
- Built-in virtual machine support

- **Drawbacks**

Limited number of supporting hardware/software

Navigation Software- Features of QNX

- **Multiprocessing (with variable priority)**

- **Scheduling**: process priorities, states and algorithms
  Priorities: 0(lowest)-31(highest)
  States: *READY, BLOCKED*
  (*send-, receive-, reply-blocked*)
  Algorithm: *FIFO, Round-robin, Adaptive*

- **IPC**
  Messages, proxies, shared memories

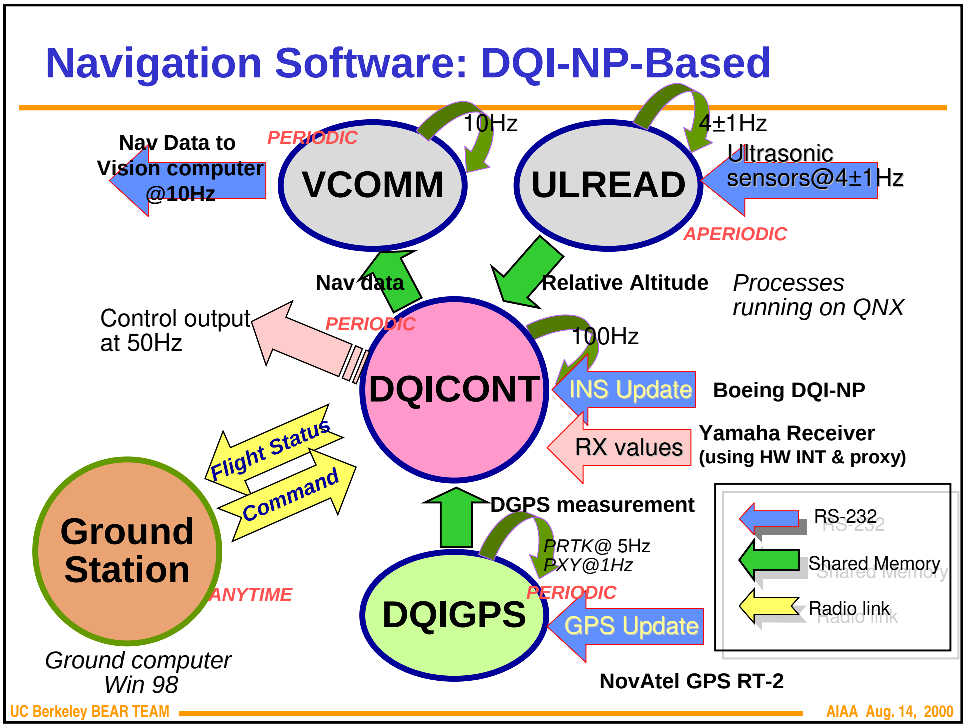
- **Real-time feature**
  supports PC hardware interrupts via *Proxy*

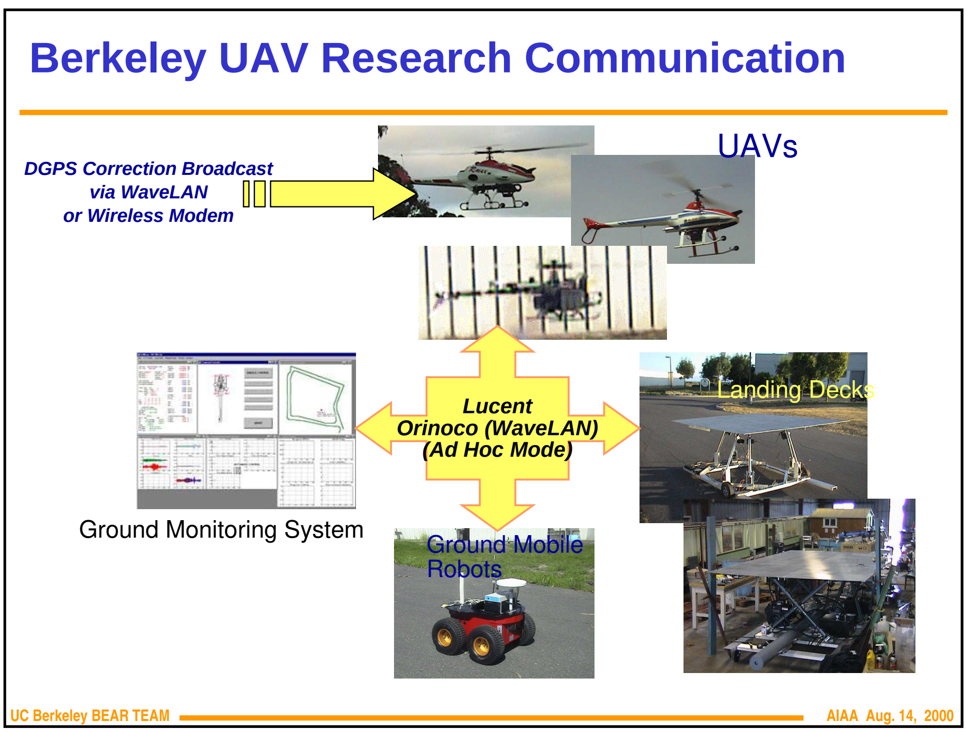
Navigation Software- Network support

Network support of QNX

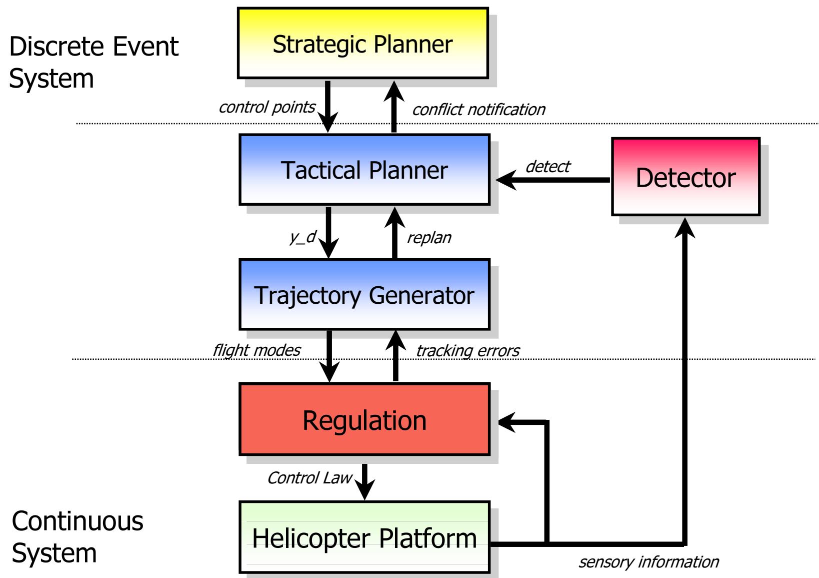
- QNX-only network ***(FLEET)***: seamless virtual machine distributed over network (supports TCP/IP, RS-232)
  supports spawning processes and IPC over network

- Heterogeneous network: QNX and non-QNX computers
  Flight info download and command upload using TCP/IP

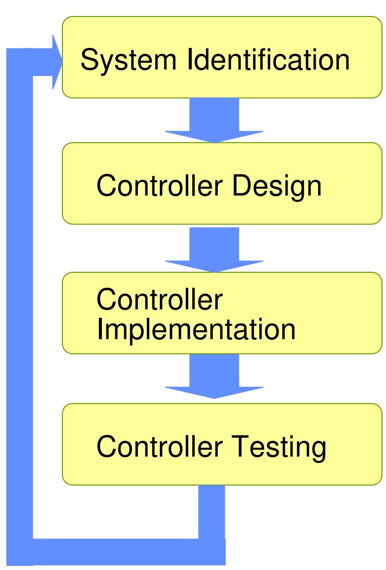

Navigation Software: DQI-NP-Based
Nav Data to Vision computer @10Hz
PERIODIC
VCOMM
10Hz
ULREAD
4±1Hz
Ultrasonic sensors@4±1Hz
APERIODIC
Nav data
Relative Altitude
Processes running on QNX
Control output at 50Hz
PERIODIC
DQICONT
100Hz
INS Update
Boeing DQI-NP
RX values
Yamaha Receiver (using HW INT & proxy)
Flight Status
Command
Ground Station
ANYTIME
DGPS measurement
PRTK@ 5Hz
PXY@1Hz
PERIODIC
DQIGPS
GPS Update
RS-232
Shared Memory
Radio link
Ground computer Win 98
NovAtel GPS RT-2
UC Berkeley BEAR TEAM
AIAA Aug. 14, 2000


Berkeley UAV Research Communication
DGPS Correction Broadcast via WaveLAN or Wireless Modem
UAVs
Lucent Orinoco (WaveLAN) (Ad Hoc Mode)
Landing Decks
Ground Monitoring System
Ground Mobile Robots
UC Berkeley BEAR TEAM
AIAA Aug. 14, 2000

Hierarchy of the UAVS Management System



UAV FCS Design: *Regulation Layer*



- Parameter acquisition (step through)
- System estimation/validation with input-output samples

- Different design methods and benchmarks
- Verify the robustness of the controller

- Code generation (Manual or Computer-aided)

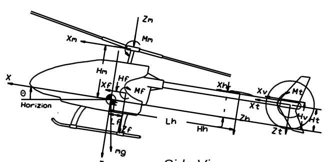
- To evaluate the controller performance
- Serious difficulties and risks involved

Flight Control System Design: Identification

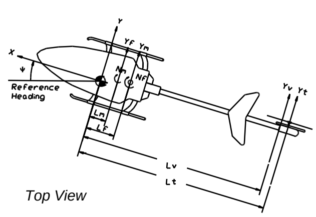
1. Aerodynamics based approach
   - use nonlinear aerodynamic equations for lumped model
   - needs extensive tests on aerodynamic components
   - still inaccurate knowledge over aerodynamic characteristics

2. Model-based parameter identification approach
   - with proper data set, high fidelity model can be acquired
   - careful experiment design and numerical process are required

RUAV Dynamics Modeling

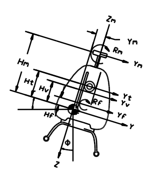


Side View



Top View

Free-body Diagram of Helicopter in flight



Rear View

Equation of Motion of helicopter rigid-body model

$$X_M + X_T + X_H + X_V + X_F = mg\sin\theta + m(\ddot{x} - \dot{y}r + \dot{z}q)$$
$$Y_M + Y_T + Y_V + Y_F = -mg\sin\phi + m(\ddot{y} - \dot{x}r - \dot{z}p)$$
$$Z_M + Z_T + Z_H + Z_V + Z_F = -mg\cos\theta + m(\ddot{z} - \dot{x}q + \dot{y}p)$$
$$R_M + Y_M h_M + Z_M y_M + Y_T h_T + Y_V h_V + Y_F h_F + R_F = I_{xx}\dot{p} - qr(I_{yy} - I_{zz})$$
$$M_M - X_M h_M + Z_M l_M + M_T - X_T h_T + Z_T h_T - X_H h_H + Z_H l_H - X_V h_V + M_F$$
$$= I_{yy}\dot{q} - pr(I_{zz} - I_{xx})$$
$$N_M - Y_M l_M - Y_T l_T - Y_V l_V + N_F - Y_F l_F = I_{zz}\dot{r} - pq(I_{xx} - I_{yy})$$

RUAV Dynamics Modeling

3. Nonlinear Model: derived from aerodynamic equations
   more potential application for controller design and simulation model
   difficult to reconcile with empirical data if extensive testing is not possible

4. Parametrized Linear Time-Invariant Model with augmented servorotor dynamics(Mettler, 1999)

$$\mathbf{x} = [u\ v\ p\ q\ \Phi\ \Theta\ a_{1s}\ b_{1s}\ \ w\ r\ r_{fb}]^T \qquad \mathbf{u} = [u_{a_{1s}}\ \ u_{b_{1s}}\ \ u_{\theta_M}\ \ u_{r_{ref}}\ ]^T$$

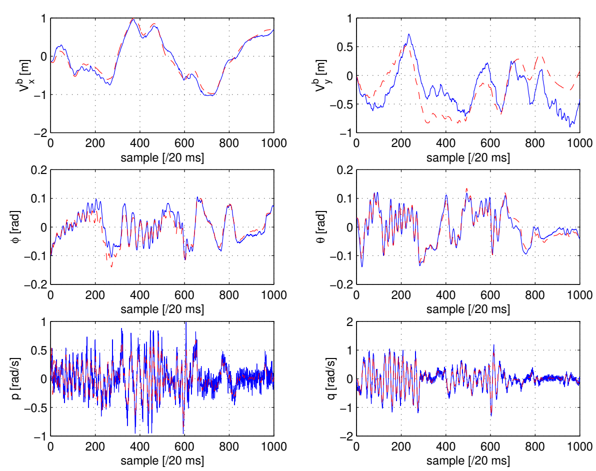
$$\mathbf{A} = \begin{bmatrix} X_u & 0 & 0 & 0 & 0 & -g & X_{a_{1s}} & 0 & 0 & 0 & 0 \\ 0 & Y_v & 0 & 0 & g & 0 & 0 & Y_{b_{1s}} & 0 & 0 & 0 \\ L_u & L_v & 0 & 0 & 0 & 0 & L_{a_{1s}} & L_{b_{1s}} & 0 & 0 & 0 \\ M_u & M_v & 0 & 0 & 0 & 0 & M_{a_{1s}} & M_{b_{1s}} & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & -1/\tau_f & A_{b_{1s}} & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & B_{a_{1s}} & -1/\tau_f & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & Z_{a_{1s}} & Z_{b_{1s}} & Z_w & Z_r & 0 \\ 0 & 0 & N_p & 0 & 0 & 0 & 0 & 0 & N_w & N_r & N_{rff} \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & K_r & K_{rfb} \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ A_{u_{a1s}} & A_{u_{b1s}} & 0 & 0 \\ B_{u_{a1s}} & B_{u_{b1s}} & 0 & 0 \\ 0 & 0 & Z_{u_{\theta M}} & 0 \\ 0 & 0 & N_{u_{\theta M}} & N_{u_{\theta T}} \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Identification result (Ursa Magna2)

- Parameters are found by PEM(Prediction Error Method) with "divide-and-conquer" approach in iterative manner

- Susceptible to fall into local minima: physical insight and any prior knowledge of the system should be applied whenever possible

- "Divide-and-Conquer" approach



RUAV Dynamics Modeling: Yamaha R-50

Identified model for *Ursa* Magna *2*

$$
\mathbf{A} = \begin{bmatrix}
-0.1257 & 0 & 0 & 0 & 0 & -g & -g & 0 & 0 & 0 & 0 \\
0 & -0.4247 & 0 & 0 & g & 0 & 0 & g & 0 & 0 & 0 \\
-0.1677 & 0.0870 & 0 & 0 & 0 & 0 & 36.7050 & 161.1087 & 0 & 0 & 0 \\
-0.0823 & -0.0518 & 0 & 0 & 0 & 0 & 63.5763 & -19.4931 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & 0 & 0 & -3.4436 & 0.8287 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 0 & 0 & 0.3611 & -3.4436 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -38.9954 & 9.6401 & -0.7598 & 8.4231 & 0 \\
0 & 0 & -1.33 & 0 & 0 & 0 & 0 & 0 & 0.0566 & -5.5105 & -44.8734 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1.8157 & -11.0210
\end{bmatrix}
$$

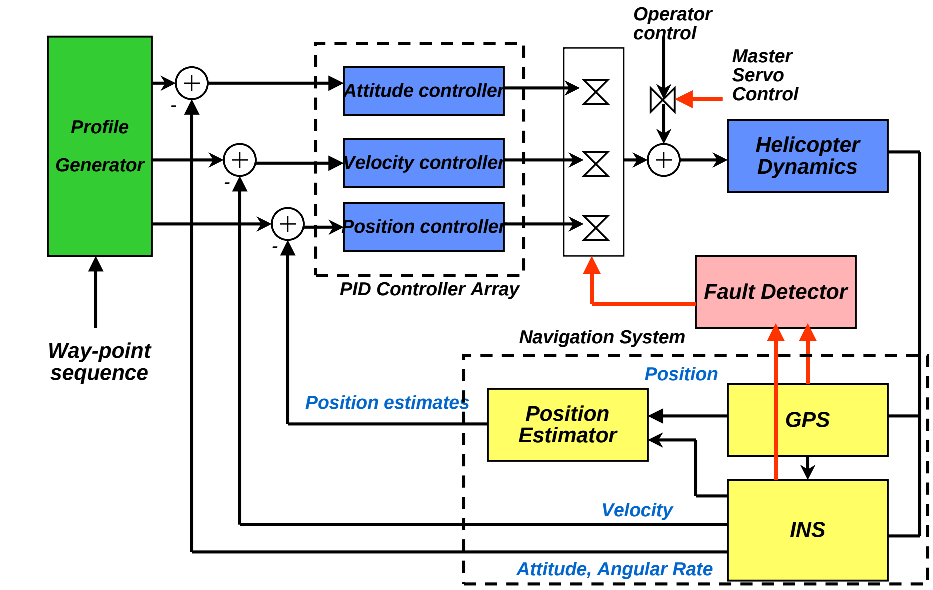
$$
\mathbf{B} = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
-0.8417 & 2.8231 & 0 & 0 \\
-2.4090 & -0.3511 & 0 & 0 \\
0 & 0 & 70.5041 & 0 \\
0 & 0 & 23.6260 & 44.8734 \\
0 & 0 & 0 & 0
\end{bmatrix}
$$

Flight Control System Design: Controller Design

1. Classical controller design
   - Design compensators for each SISO loop of roll/Vy, pitch/Vx, Yaw, Heave channel
   - Simple structure, easy to design, low computing load
   - Flexible to introduce fault tolerance algorithm

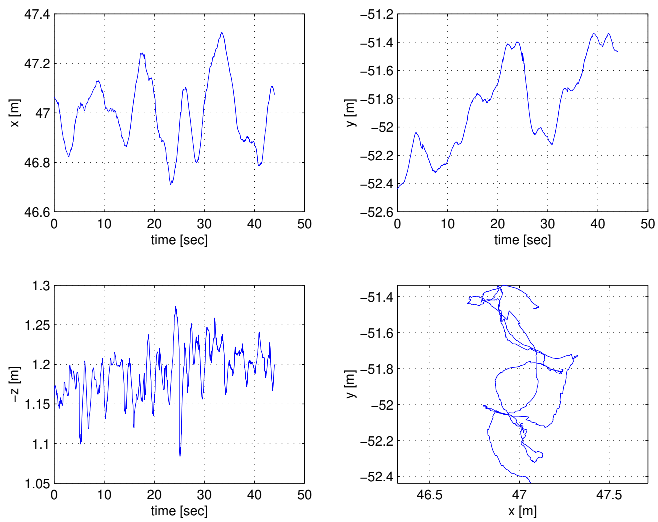
2. Advanced control theory *(Shim et al, CDC 1998, Shim, CDC 2000)*
   - Linear robust control theory
     mu-synthesis: very high order.
     Effective to imperfect knowledge over the system
   - Fuzzy control with genetic algorithm
   - Feedback linearization

Flight Control System: SISO Controller Structure

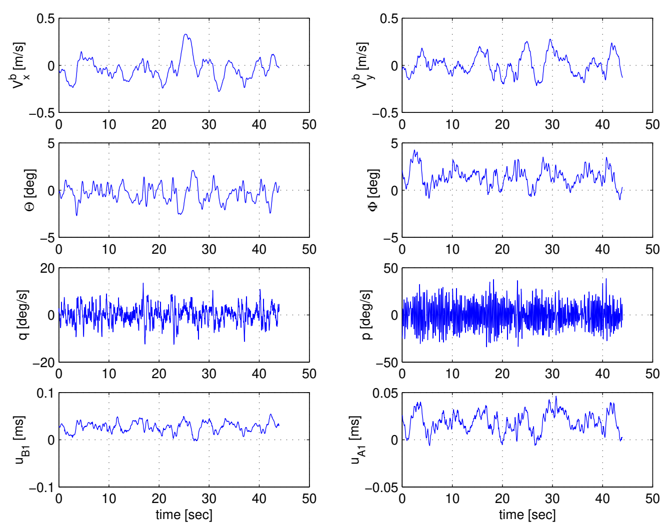


Flight Control System Experiment: (May 16, 2000) Attitude-Velocity-Position-Heading Control

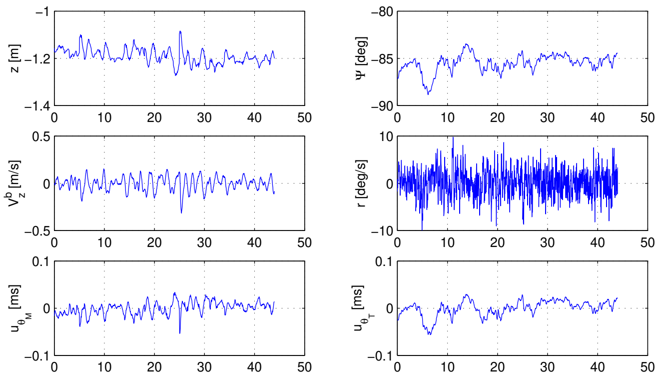
Flight Control System Experiment: Attitude-Velocity-Position Control (Aug 6, 2000)



Flight Control System Experiment: Attitude-Velocity-Position Control (Aug 6, 2000)



Flight Control System Experiment: Attitude-Velocity-Position Control (Aug 6, 2000)

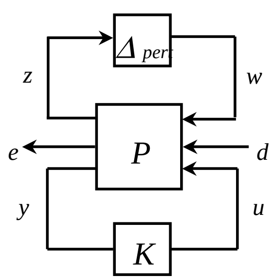


μ-Synthesis Application

Control Objective:

Find a stabilizing controller K, *such that for all such perturbations* Δ_{pert}*,* *the closed-loop system is stable and satisfies*

$$\|T_{ed}\|_\infty = \|F_L[F_U(P,\Delta_{pert}),K]\|_\infty \le 1$$



LFT representation

$$P(s) = \left[\begin{array}{c|ccc} A & B_1 & B_2 & B_3 \\ \hline C_1 & D_{11} & D_{12} & D_{13} \\ C_2 & D_{21} & D_{22} & D_{23} \\ C_3 & D_{31} & D_{32} & D_{33} \end{array}\right]$$

μ-Synthesis Application

- Uncertainty

  unstructured multiplicative uncertainty at input channel

$$W_U = 0.1\frac{s+1}{s/10+1}$$

- Noise Weighting

  realistic noise model based on the FFT result of INS sensor

- Actuator Weighting

  limits the control input from saturation

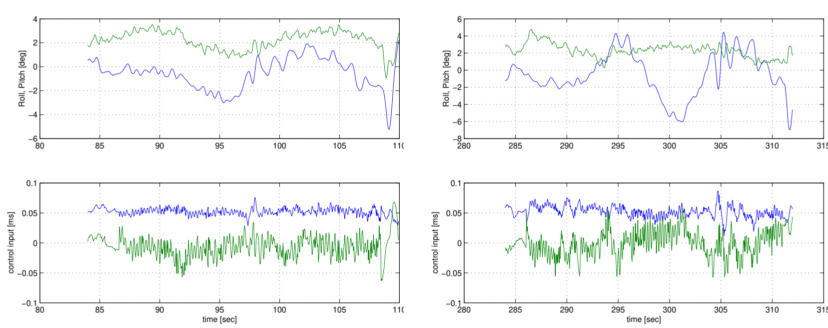
$$W_A = \frac{s/20+1}{s/50+1}$$

- Handling Quality Model/Performance Weighting

$$W_Q = \frac{\omega_n^2}{s^2 + \omega_n s + \omega_n^2}, \quad \omega_n = 0.6\pi, \; W_{\Phi,\Theta} = 100\frac{s/100+1}{s+1}, \; W_{P2} = 3\frac{s+1}{s/0.2+1}$$

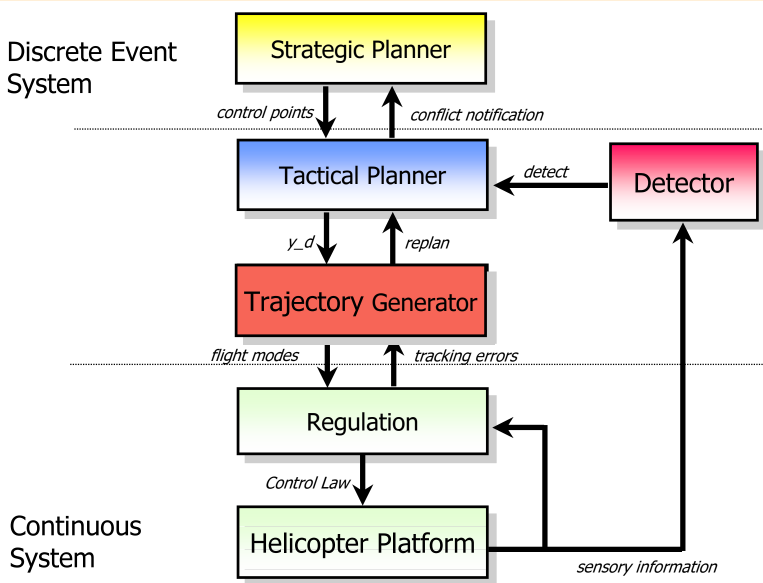
μ-Synthesis Controller Test

- Attitude(Roll/Pitch) controller designed and tested
- Rate gyro noise spectrum is found and used for noise model
- Stable regulation was shown
- Needs to be extended to linear velocity control
- Z/Yaw controller can be designed separately due to the weak coupling between longitudinal and lateral/vertical/yaw dynamics



μ-Synthesis Attitude Controller Test (July 6, 2000)

Hierarchy of the UAVS Management System



Vehicle Control Language

Objective

Develop an abstract and unified UAV mission control language environment

Features

- Mission-independent
- Executed as batch or interactive mode
- Seamlessly integrated with existing hierarchy
- Can be integrated with graphic interface via automatic code generator

Vehicle Control Language

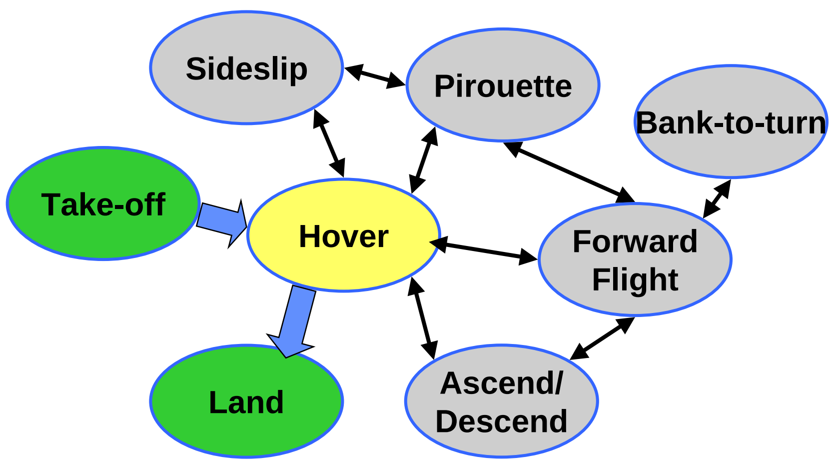
Motivation

- Vehicle control programming is based on proprietary methodologies
- Motion control programming is dependent on the mission
- Not compatible with other methods

Conclusion: a generalized, vehicle-independent, mission-independent descriptive language is required.

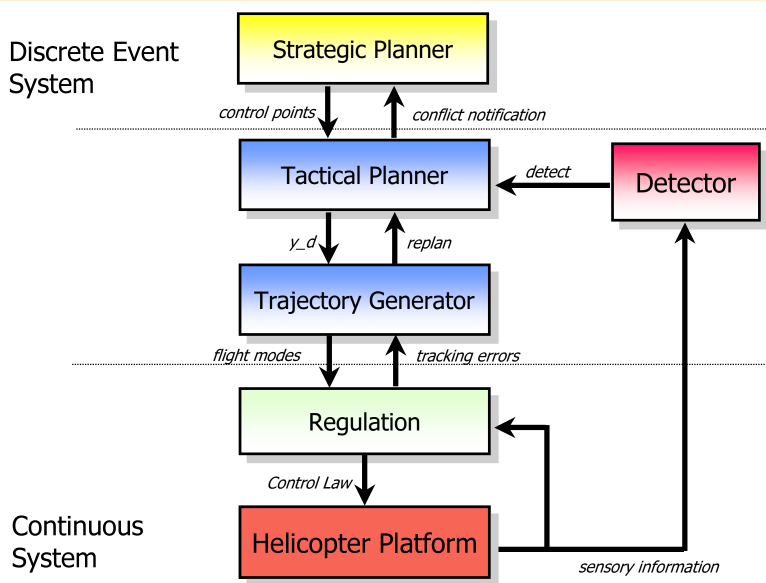
Flight control synthesis: way-point navigation

Helicopter Mode transition



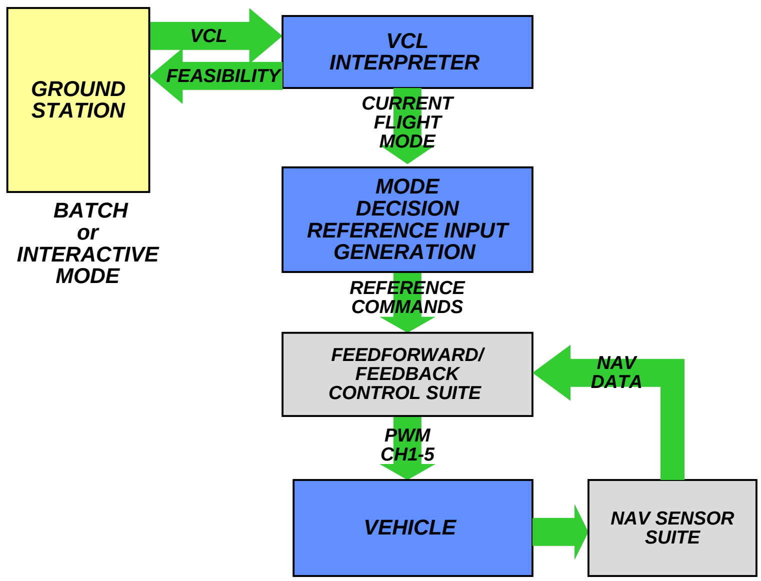
UC Berkeley BEAR TEAM   AIAA Aug. 14, 2000

Hierarchy of the UAVS Management System



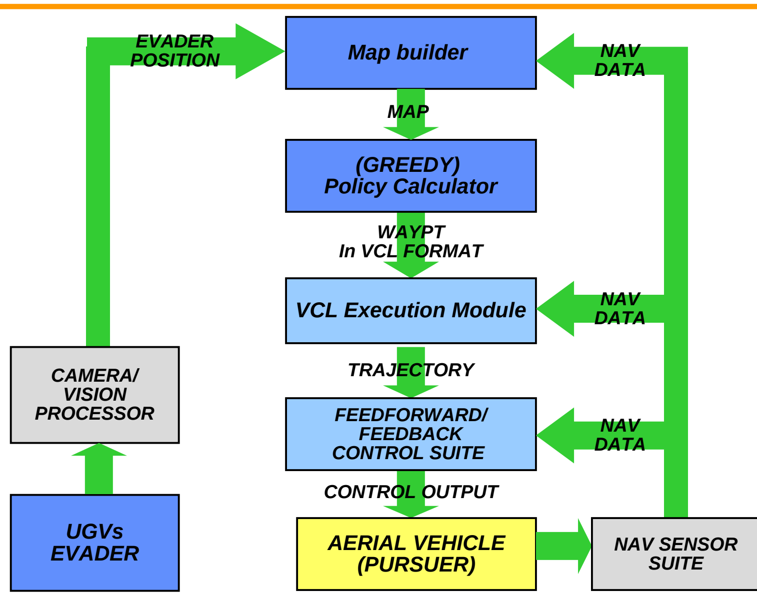
UC Berkeley BEAR TEAM   AIAA Aug. 14, 2000

VCL Execution Module Structure



UC Berkeley BEAR TEAM AIAA Aug. 14, 2000

VCL Application for Pursuit-Evasion Game Execution Module Structure



UC Berkeley BEAR TEAM AIAA Aug. 14, 2000