Planning, Execution & Learning: Monitoring and Diagnosis

Reid Simmons

Why Monitor?

- Detect Internal Faults
  - Hardware failure
  - Software errors
- Detect Unexpected Contingencies
  - Changes in environment
  - Actions not going according to plan
- Detect Unexpected Opportunities
- Compensate for Incomplete Policies
  - Behaviors not available for ***every*** state

Terminology

- **Expectation**: Anticipated Future State of the World
- **Exception**: Violated Expectation
  - Divergence between predicted state and observations
- **Monitoring**: Detect Exceptions
- **Diagnosis**: Isolate Fault From Symptoms
- **Recovery**: Bring Plan into Alignment with Observations

Approaches

#1 Fault Models

– Explicitly enumerate fault modes
– One-to-one correspondence between fault mode and fault
+ Diagnosis is easy
- Hard to anticipate all possibilities

#2 Expectation-Based

– Compare model of expected behavior against observations
– Trace back from symptoms to find faulty components
+ Easier to specify “nominal” behaviors
- Diagnosis is hard (and often ambiguous)

Approaches are not inconsistent: May be combined

Progress-Based Approach (Simmons)

- Deals with ***Unanticipated*** Exceptions
- Track ***Progress*** Towards Goal
  - Lack of progress/slower progress than expected
- Maintain Hierarchy of Monitors
  - Detect exceptions at different temporal scales
  - More general monitors handle wider range of situations
  - More specific monitors trigger sooner and impart more diagnostic information

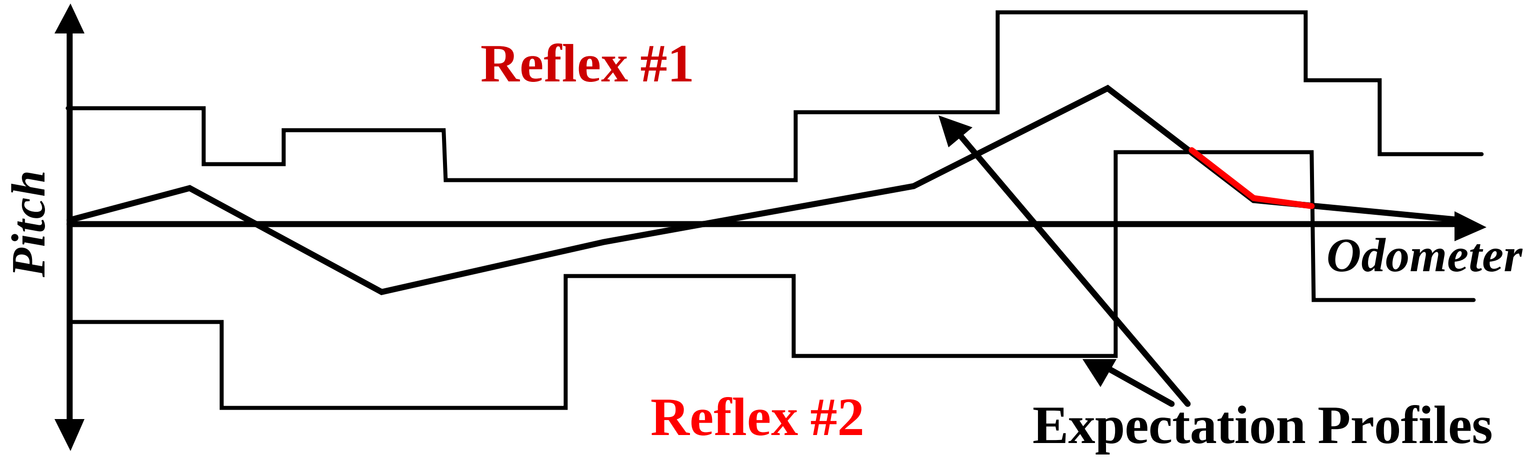
Monitoring Xavier's Navigation

- Goal: **Navigate to location X while avoiding obstacles**

- Expectations for Progressing Towards Goal
  - ***Time-Out***: Robot should reach goal *K* standard deviations after average travel time (based on path)
  - ***Position***: Deviation between predicted position (based on path) and observed (most likely) position should not increase "too fast"
  - ***Looping***: Robot should not return to a given state, traveling in the opposite direction (detect cycles in POMDP navigation)
  - ***Spinning***: Robot should not oscillate in one place for "too long"

Profile-Based Approach (Miller)

- Dynamic Creation of Expectations
  - Simulate plan
  - Record temporal profile of sensor values
  - Account for uncertainty (actuator, sensor, environment)

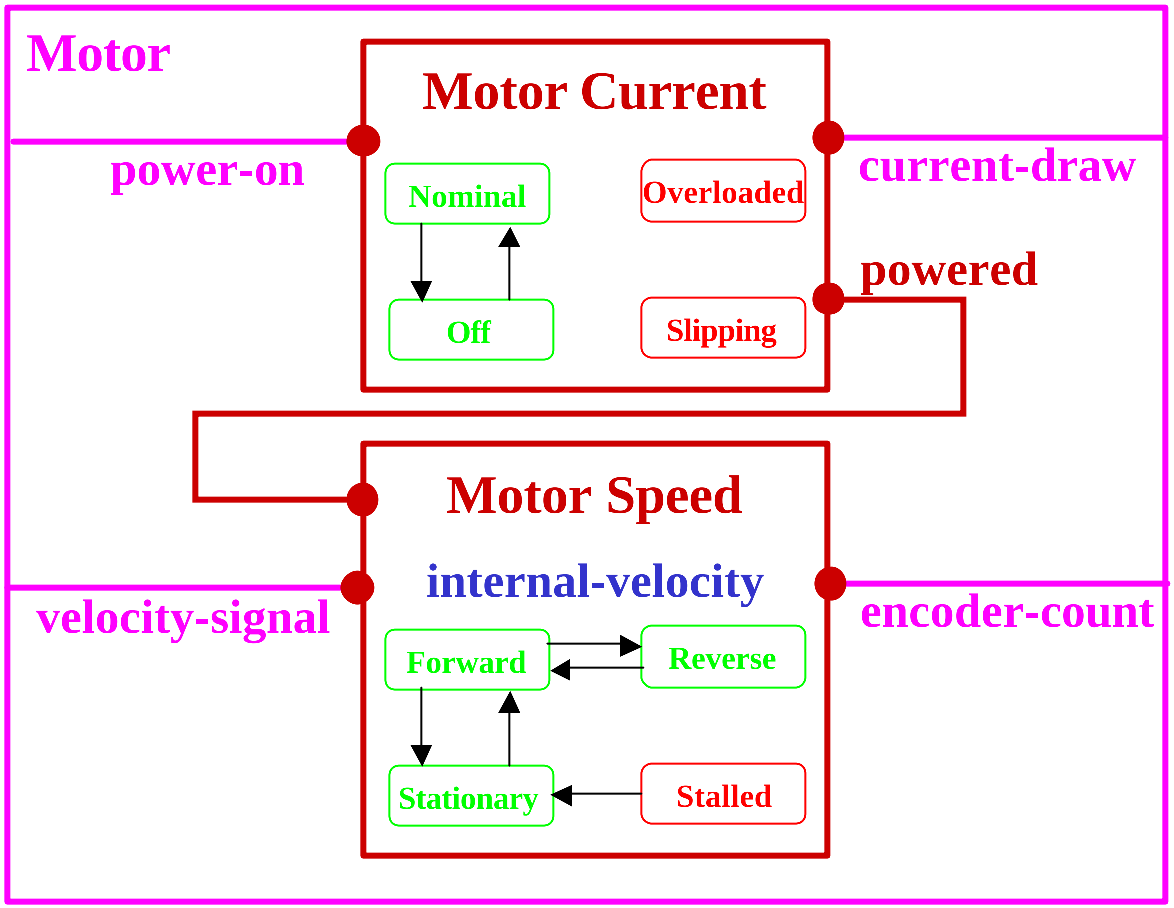


- Monitor Expectations for Each Sensor
  - Associate reflex action with profile violations

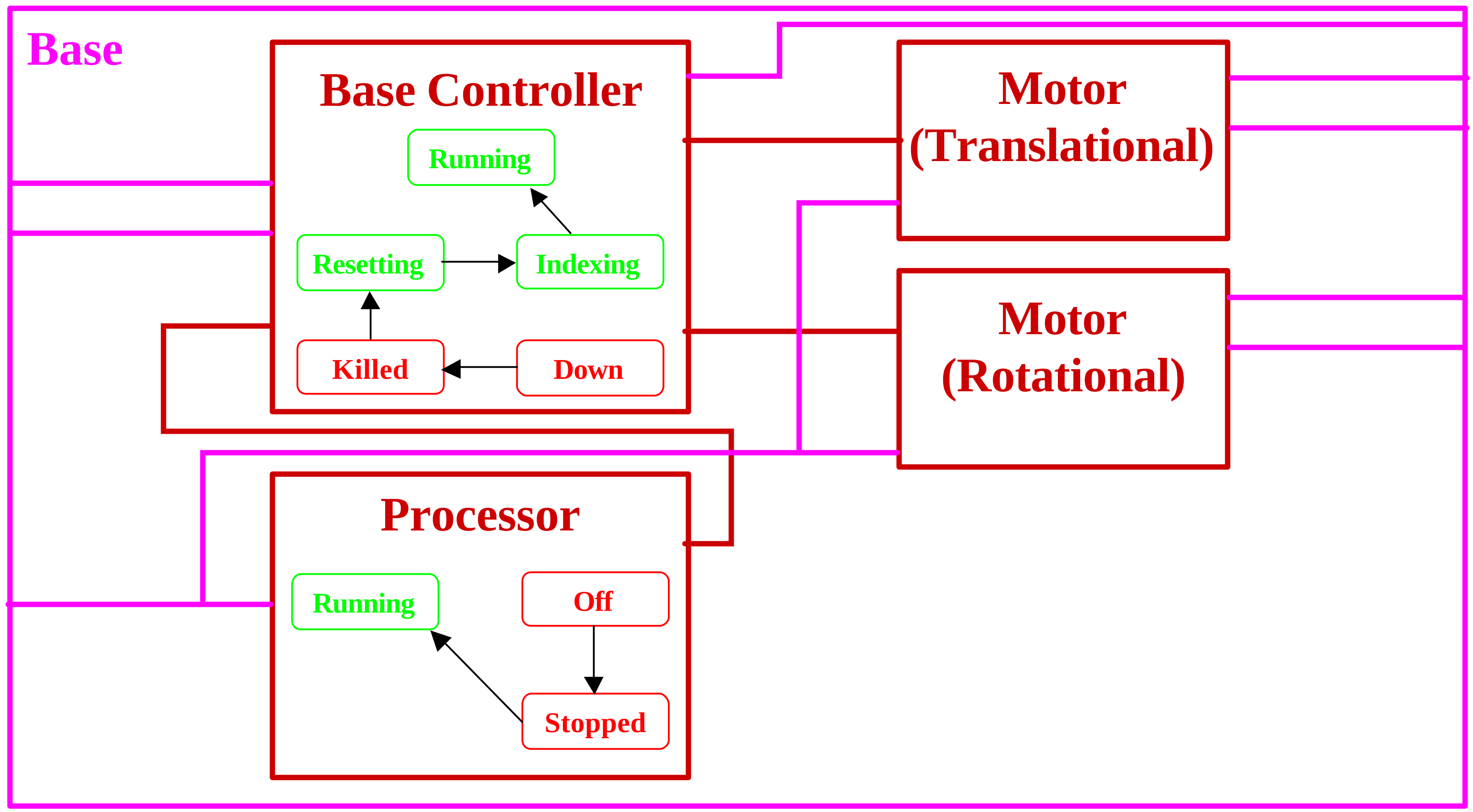
Livingstone (Williams & Nayak)

- Developed at NASA Ames
  - Used on Remote Agent for Mode Identification and Recovery
- Based on Symbolic, Qualitative Models
  - State transition diagrams (nominal and fault modes)
  - Inter-connections between components
  - Propositional relationships between variables
- Approach
  - Use models and commanded inputs to generate predications
  - Detect inconsistencies between predictions and observations
  - Find "***conflict set***" of components whose malfunction can explain discrepancy
    - Uses very efficient "Truth Maintenance System"

Xavier Component Models



Xavier Component Models



Model-Based Monitoring of Xavier

- Dealing with Observations
  - Transform sensor readings (e.g., encoder counts, velocities) into qualitative values (negative, zero, positive, small, large)
  - May be context dependent (get context from model)
  - May need to be learned
- Dealing with Commands
  - Predict state transitions based on behavior commands
  - Need to take command latency into account
- Integrates Easily into Publish/Subscribe Architecture
- Runs in Real Time (in Lisp!) On-Board the Robot

Monitoring Hybrid Systems

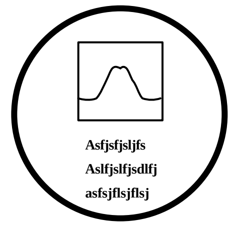
- Combines Continuous and Discrete Dynamics
  - Discrete mode depends on continuous state
  - Continuous dynamics depends on mode

- Problem is Monitoring in Face of Uncertainty
  - Often cannot directly observe mode or continuous state

- Approaches
  - Track most likely state (*Livingstone*)
  - Discretize continuous state and track using POMDP(*Fernandez*)
  - Approximate continuous state (*Washington*)
  - Approximate belief state (*Verma*)

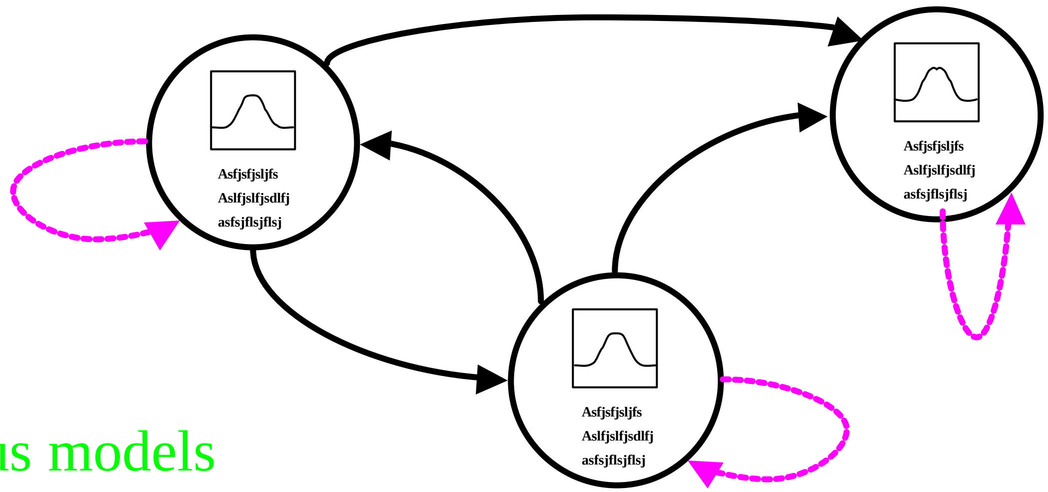
Markov & Kalman Models (Washington)

- Representation
  - Represent continuous state using bank of Kalman filters
  - Represent discrete mode using POMDP



- Estimation
  - Each mode is associated with different KF model
    - Different constraints; Different gains
  - KF used to estimate observation probabilities for POMDP
    - $p(o \mid s) \approx p(o \mid KF) \bullet p(KF \mid s)$

Markov & Kalman Models

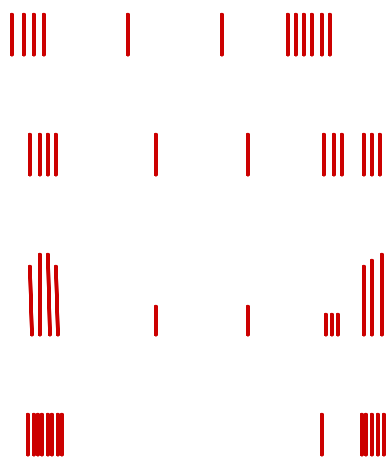


- Pros
  - \+ Simple continuous models
  - \+ Computationally very efficient
  - \+ Captures hybrid dynamics
- Cons
  - – Noise may not be Gaussian
  - – Evolution of Kalman Filters depend on initial conditions, which in turn depend on when discrete state is entered
    - Limit number of filters

Particle-Filter Based Approach (Verma)

- Representation
  - Represent complete continuous and discrete state
  - Represent complete transition and observation probabilities
  - Approximate belief state using samples (***Particle Filter***)

- Estimation
  - Update samples according to transition probabilities
  - Reweight according to observation probabilities
  - Resample based on weightings



Particle-Filter Based Approach

- Pros
  - + Can use high fidelity prediction models
  - + Non-parametric probability distribution
  - + Near-constant time computation (independent of size of state space)
- Cons
  - – Does not track low probability events well
    - Sample from mixture of prior and observation distributions
    - Sample from mixture of prior and utility (loss)
      - – Focuses on high-risk parts of state space