Control/Learning Architectures For Use in Robots Operating in Unstructured Environments

October 2, 2000

Patrick McDowell

S.S. Iyengar

Marlin Gendron

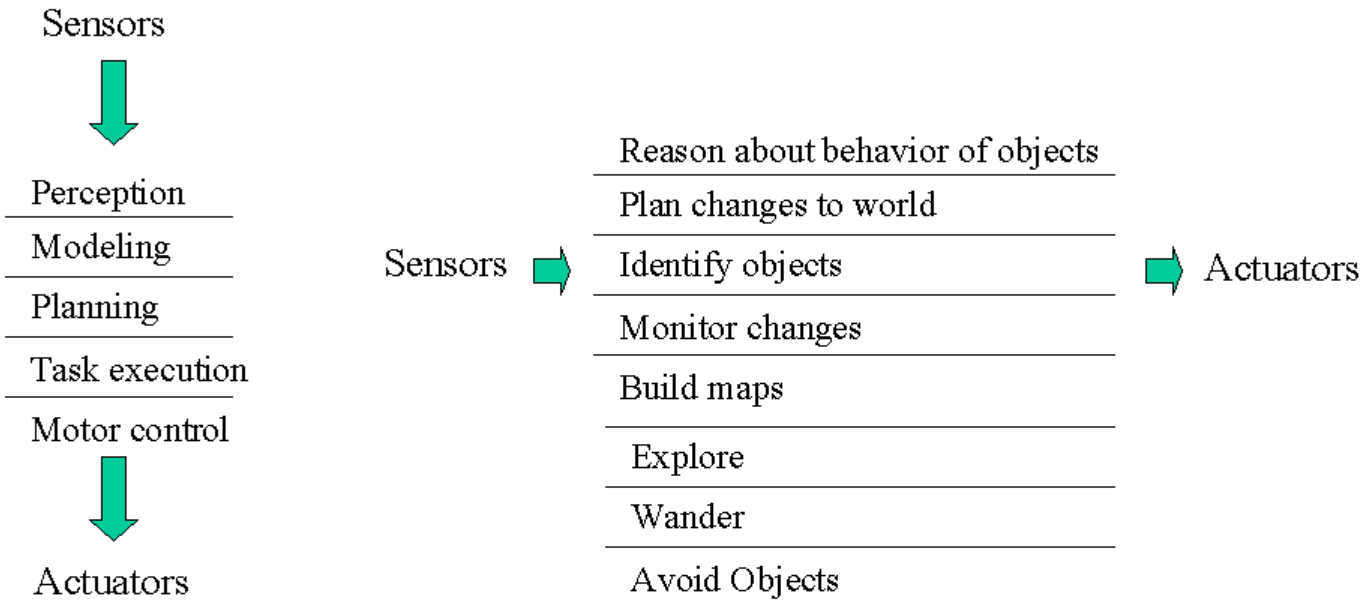
Brian Bourgeois

John Sample

Introduction

- The work presented in this paper originated from the LSU Robo-Tiger project.
- The idea was conceived by Dr. S.S. Iyengar and Dr. Lynn Jelinski late in the fall semester of 1999.
- The goal was to build a robotic tiger that would provide a research platform for mobile robots in unstructured environments, and catch the imagination of the students and faculty.
- We present some of the issues and results from the small prototype phase of the project.

Data/Control flow in Subsumption vs. Classical AI



This figure shows the difference between functional and behavioral decompositions. On the left, the figure shows the traditional decomposition of a mobile robot into functional modules. On the right, the figure shows a decomposition of a mobile robot control system based on task achieving behaviors.

Learning Using Subsumption

- Learning under the Subsumption Architecture is primarily a matter of conflict resolution among the various behaviors, that is, deciding which behaviors should be active at any particular instance in time.
  - Behavior conflicts were resolved by the developers at compile time [Brooks, 1986] or by using a description of the desired behavior selection [Rosenschein & Kaebling, 1986]. In both cases, the behavior hierarchy is static.
  - Maes [1990] describes an algorithm which allows a behavior based robot to learn based on positive and negative feedback, thus eliminating the need for the programmer to solve the problem ahead of time.

In Figure 1, notice that the body proportions of Mickey are similar to that of a cat. Figure 2 shows that Stubby is cat size, but in Figure 3 it's simplified joint structure is apparent.

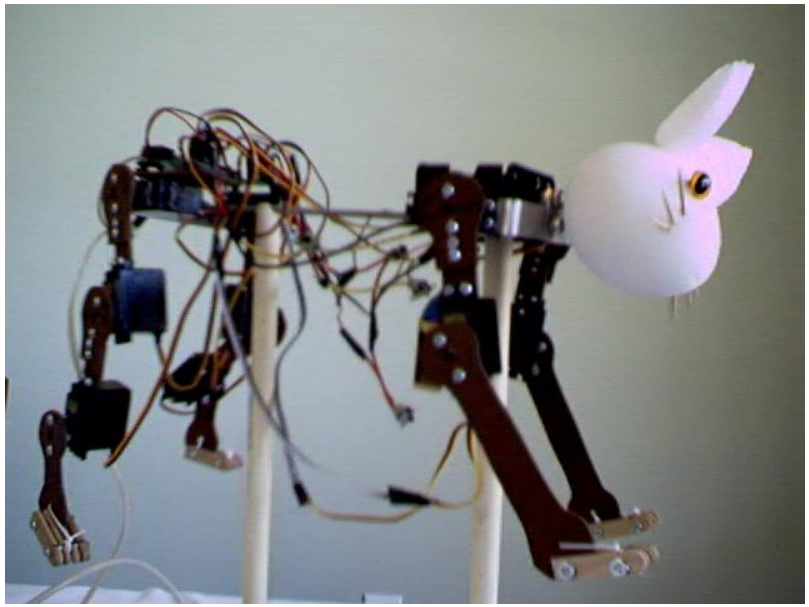


Figure 1. Mickey on the test stand.

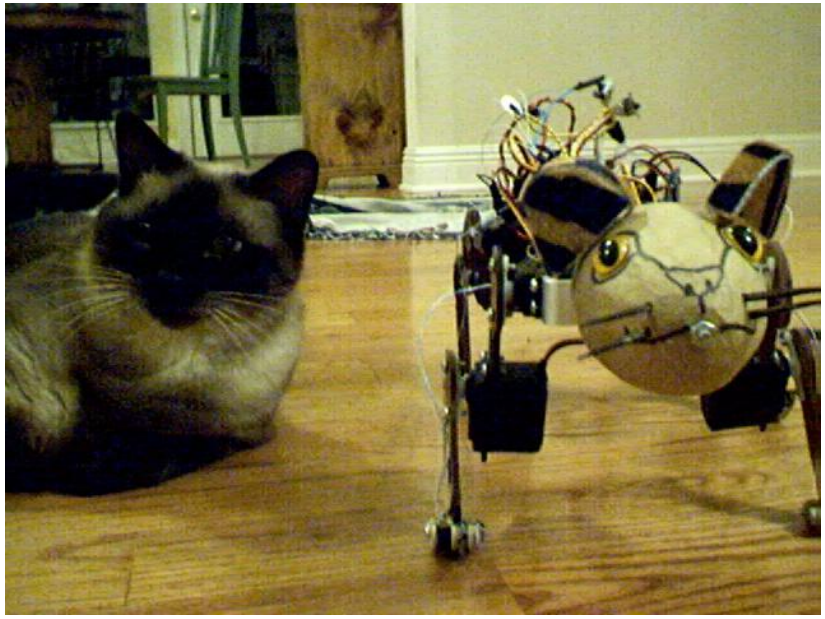


Figure 2. Mitzy on left, Stubby on the right.

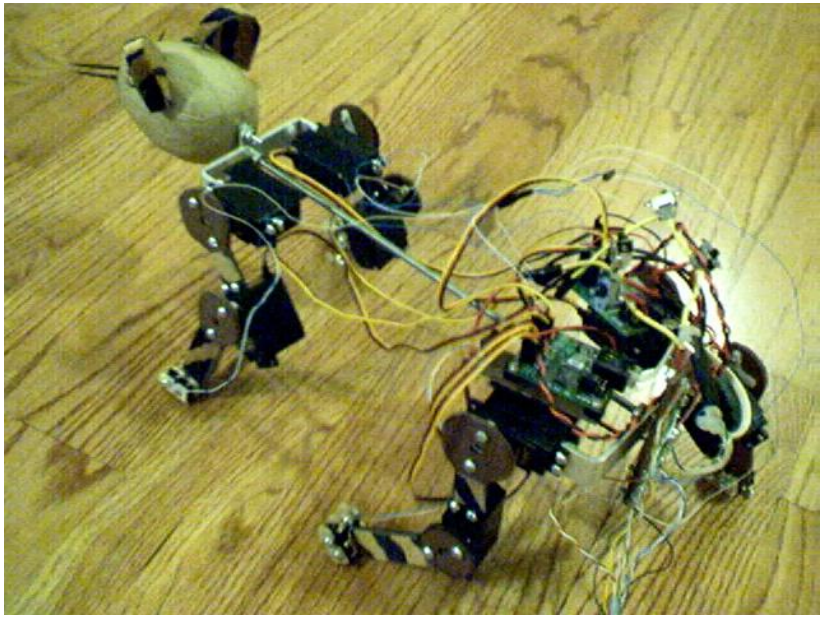


Figure 3. Top view of Stubby.

Hardware and Software (Cont.)

- 3 layers of software.
  - Gait level.
  - Command generation level.
  - Device level.

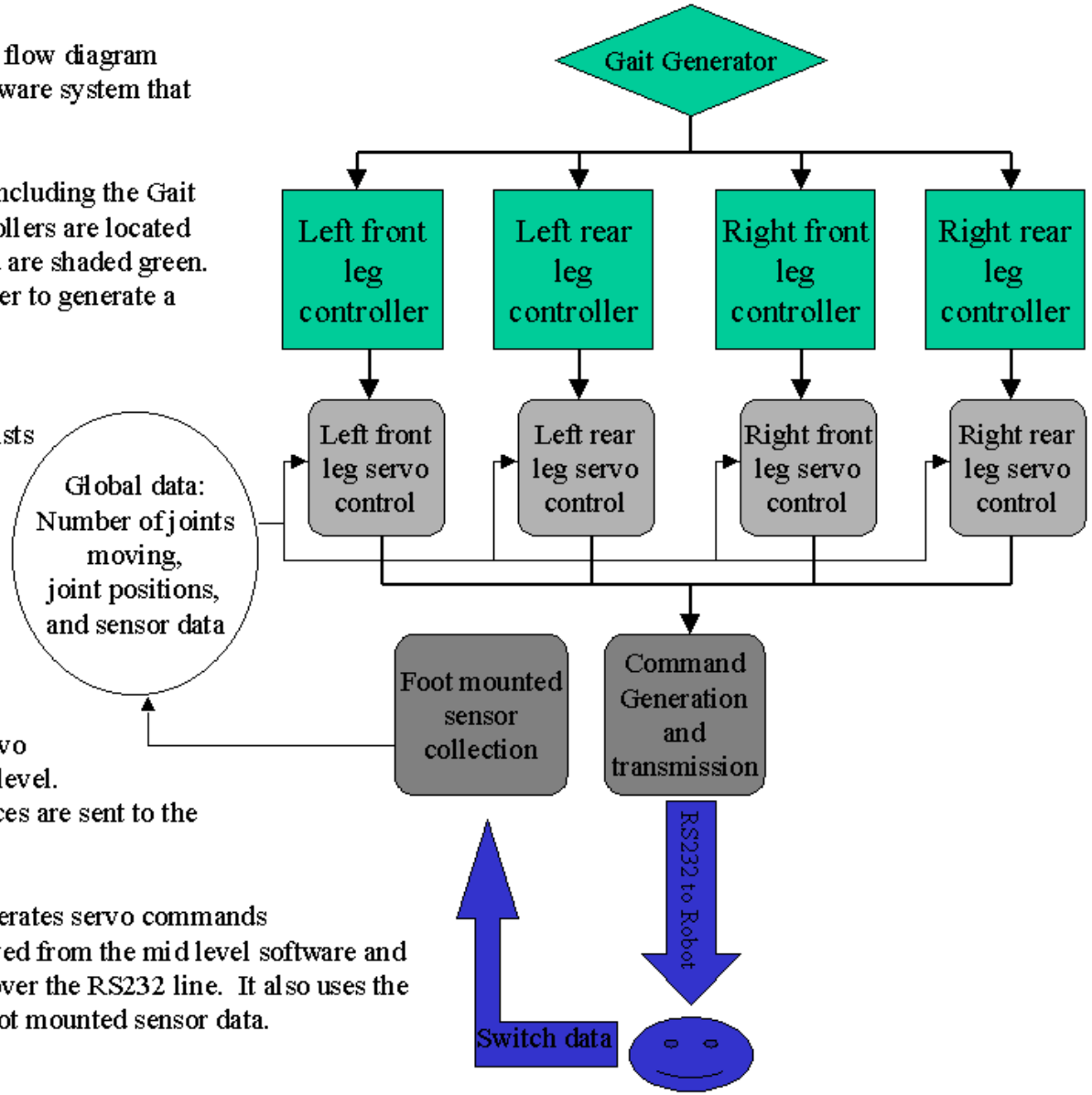
Hardware and Software (cont.)
System Overview

This figure shows an overall flow diagram for the virtual controller software system that controls Stubby.

The higher level functions, including the Gait Generator, and the leg controllers are located at the top of the diagram and are shaded green. These functions work together to generate a walking gait which is sent to the mid level software.

The mid level software consists of the leg servo controls. They are shaded light gray in the diagram. This set of modules translates the gait level data to servo position sequences. Also, at this level, all servo positions are tracked and servo speed is kept at a consistent level. The servo command sequences are sent to the device control level.

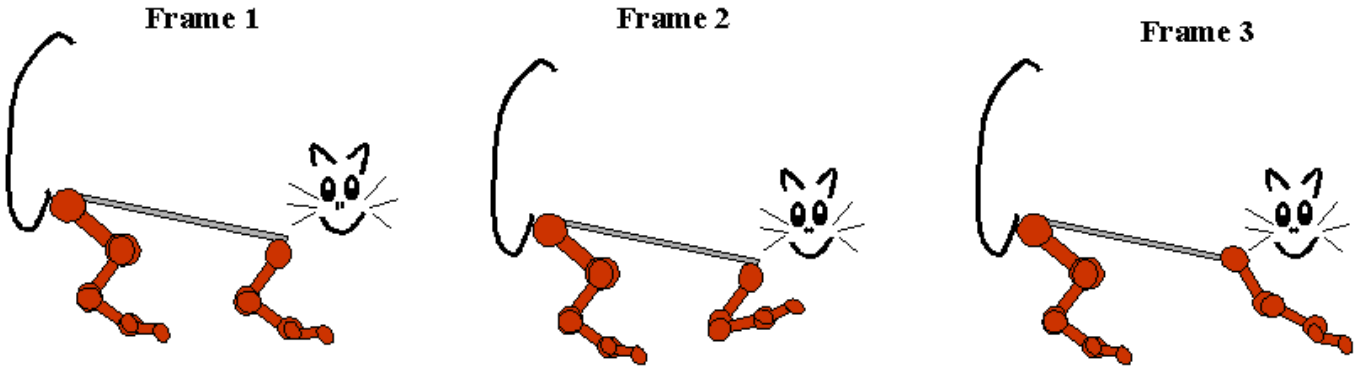
The device control level generates servo commands based on the positions received from the mid level software and transmits them to the robot over the RS232 line. It also uses the DAQCard1200 to collect foot mounted sensor data.



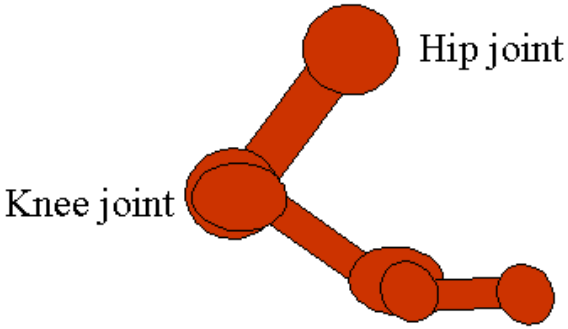
Gait level control algorithm development.

- Movie Frame method.
  - The first and most basic is the *Movie Frame* method.
  - A series of “snapshots” of the joint positions during walking are created.
  - Paradox. In order to take the snapshots, the robot first has to walk.
  - By observing cats and using a servo control program the frames were estimated and recorded. Then they were played back into the robot in a loop.
  - The typical number of frames for the loop was 8 frames.
  - Results? Less than stellar. At few staggering steps forward, but proved that the robot could physically do it.

Movie Frame Method



This sequence of images illustrates the ***frame*** concept. In this sequence the cat reaches forward with its right front leg. Frame 1 shows the cat standing. Frame 2 shows the cat lifting its right front leg. Frame 3 shows the cat setting its foot on the ground. This sequence of three frames is a **reach sequence**. During walking, each leg of the cat reaches, as shown in the sequence, and once the leg is in position it pulls with a **pull sequence**.



For each frame in the sequence the hip and knee joints have coordinates. The table below shows a possible set of coordinates for the hip and knee joints for the **reach sequence**.

Frame #	Hip joint coordinate	Knee joint coordinate
1	**10**	**20**
2	**10**	**40**
3	**20**	**10**

Command sequences are formed by generating the joint positions between the frames. For example, the knee joint in frame 1 is at position 20, and in frame 2 it supposed to be at position 40. The command sequence generator will send positions 21 to 40 to the robot over the RS232 line.

Genetic Algorithm Method

- Used to refine the best Movie Frame walking sequence.
- Since the robot is a mechanical entity, the population size and number of generations were kept low by genetic algorithm standards.
- A typical population size would be 5, and during run, 50 generations would be the outer limit.
- The fitness function was distance walked by the robot in 16 steps, measured by a yardstick.
- Each frame was a chromosome, crossover and mutation were implemented, as was sexual reproduction.

Walking Ugly

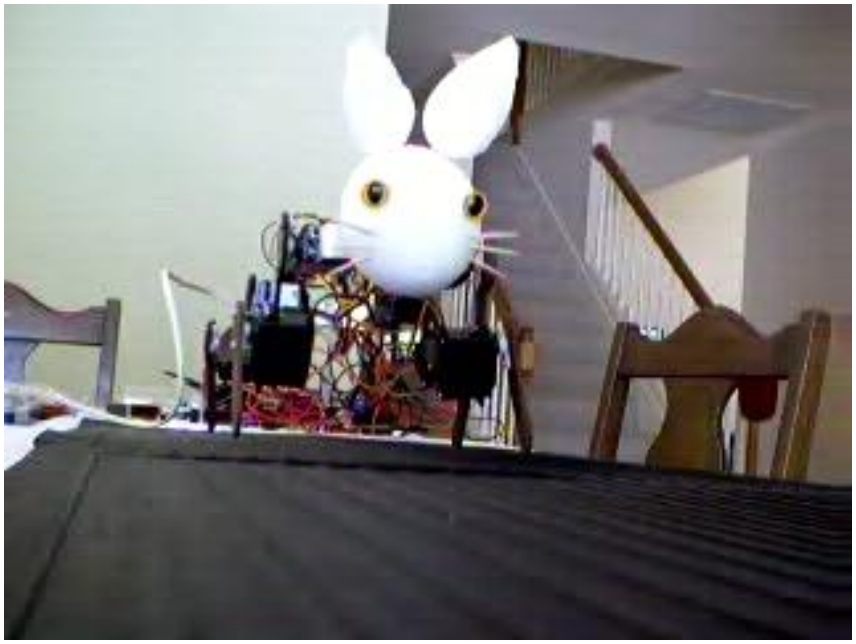


Figure 1. Stubby goes for a walk while the genetic algorithm is being run.

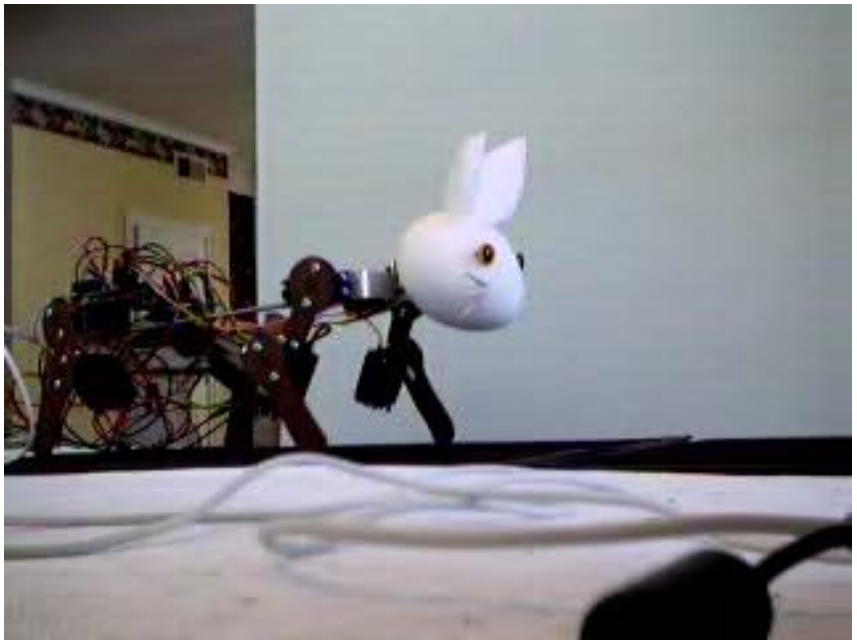


Figure 2. Stubby struggles forward during a testing session.

Virtual Controller Method

- Given a stable robot and acceptable leg reach and leg pull movements, the walking problem boils down to one of timing.
- That is, for the robot to walk, each leg has to reach and pull at the appropriate time.
- This idea is fundamentally different from the frame approach in that each leg has its own state, rather than the whole robot has its state.
  - A virtual controller per leg, implemented in LabView, is controlled by a gait generator.
  - Each virtual controller has a “reach” and “pull” subsection.
  - The controllers run asynchronously, each one sending its leg position commands to the robot over a common RS 232 line.

Virtual Controller Method (Cont.)

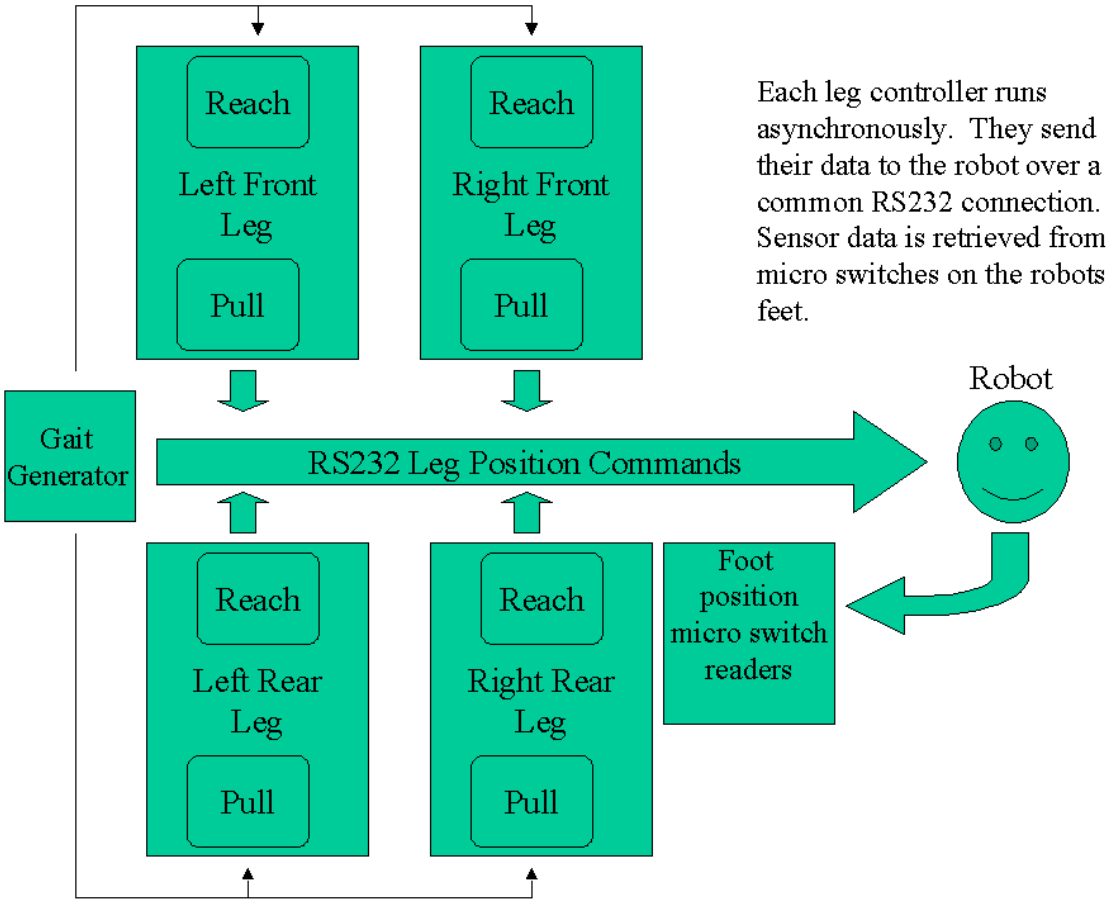


Figure 1. This figure shows a block diagram of the virtual controller system used to control Stubby. Since it is an asynchronous system, each of the processes (represented by the square boxes) is running independent of the others. The gait generator signals when the leg processes start. The leg processes signal the gait generator when they complete. The sensor reading process makes current micro switch data available to any processes in which it is required. It is implemented in LabView.

Virtual Controller Method (Cont.)

- Results? Much smoother and more efficient.
- Surface dependent. A slippery surface, like a wood floor, helps mask errors.
- Timing between steps, and the timing between the frames that the steps (movie frames) are made of influence gait effectiveness.
- MIT's virtual biped gaits tested. Trotting and galloping successful.

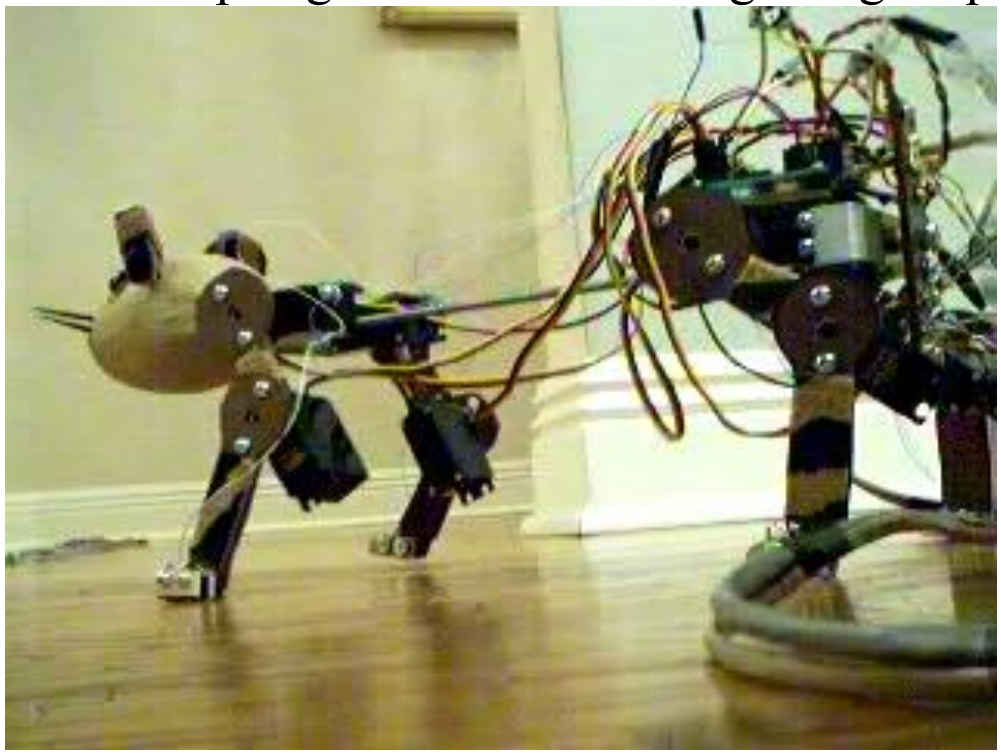


Figure 1. One of the first successful walks.

Continuing Development

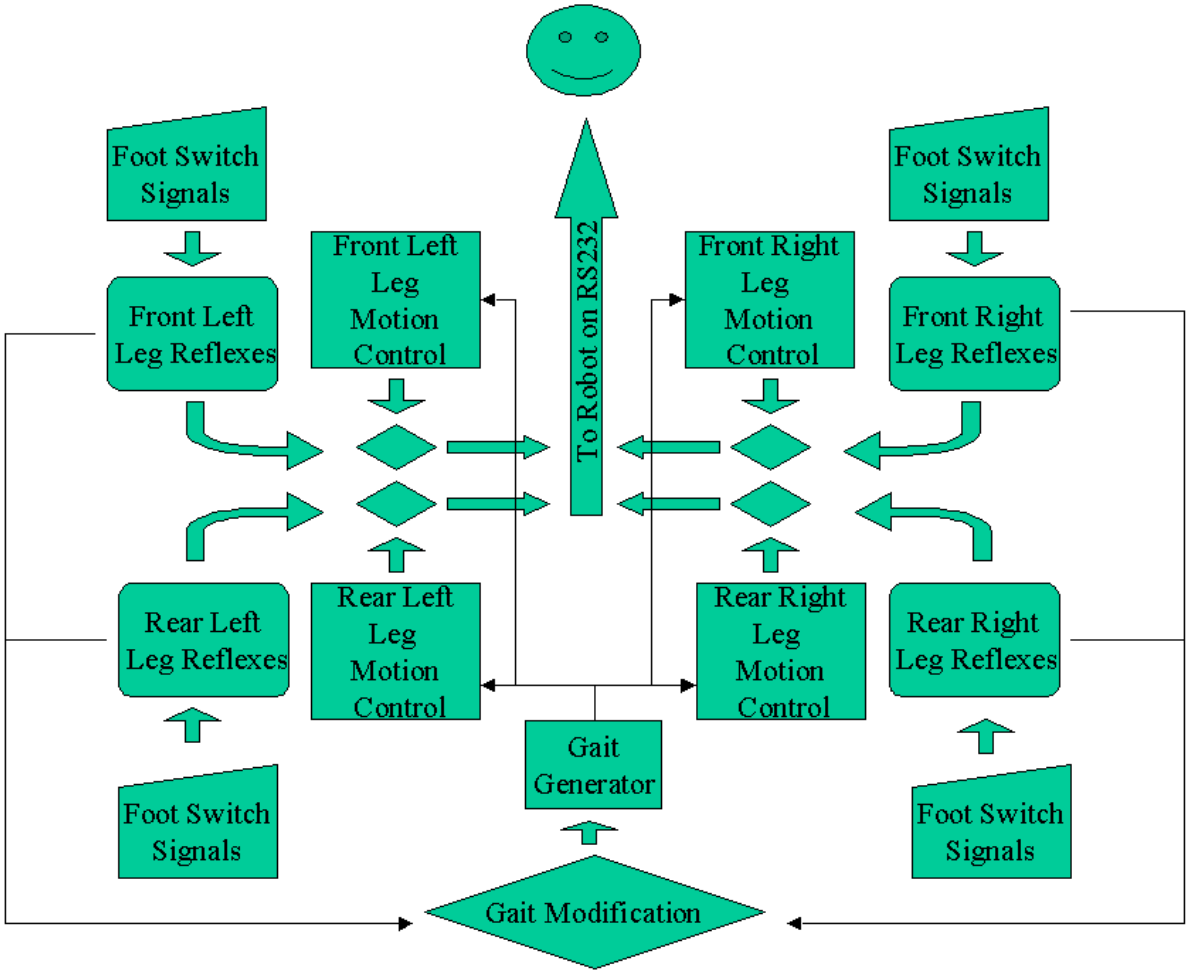


Figure 9. This diagram shows the interaction of the reflexive actions and the leg motion controls. Reflexive actions will take priority over the leg motion controls. Initially, Gait adjustment will be implemented at the Gait Generator Level, as shown, but will later be at the leg motion control level.

Summary

- Mobile robotics is an interesting and challenging discipline.
- The LSU robotic tiger project has provided a platform from which to research, explore and integrate the various technologies that will be used in creating truly useful machines that will be able to operate autonomously in unstructured environments.
- The work that we have done so far has benefited from work done by many researchers, notably those at MIT and McGill University.
- The work we planned for the near term will focus on improving performance by adding reflexive behaviors, gait optimization, and situational memories based on reflexive action driven associative memories.

Outline

- Background - Schools doing similar work, discussion some of their ideas and results.
  - MIT's AI Lab and Subsumption Architecture.
  - Learning under Subsumpiton.
  - MIT's Leg Lab and the "virtual biped".
  - McGill University and their four legged small robots.
- Hardware and Software Design the third prototype, "Stubby".
  - Basic electrical and mechanical design of Stubby.
  - Differences between Stubby and robots from MIT and McGill.

Outline (Cont.)

- System Overview
- Gait level control algorithm development
  - Movie Frame method
  - Genetic Algorithm method
  - Virtual Controller method
- Continuing Development
  - Reflexive actions.
  - Gait adjustment based on reflexive actions.
  - Gait recall using associative memory.
- Summary

Background

- MIT's AI Lab and Dr. Rodney Brooks have developed a mobile robot control paradigm called Subsumption Architecture.
  - It can be described as a hierarchical, asynchronous, distributed system whose components produce behaviors. The combination of the behaviors produce functionality.
  - Sensor input and actuator actions are closely coupled.
  - The behavioral components are set up in hierarchy of layers, in which higher behaviors can inhibit or subsume lower behaviors, thus the name Subsumption Architecture.

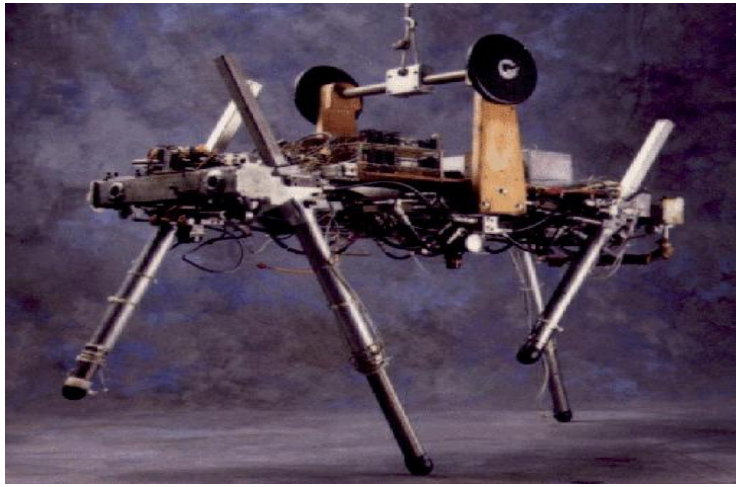
Background (Cont.)

- This technique was used to program “Genghis” a six legged robot. It was able to learn the tripod gait, using swing forward behaviors for each leg, negative feedback when it fell, and positive feedback when it moved forward.
- This is not a trivial task, neither the negative nor the positive feedback is associated with any single behavior, but with the coordinating of the behaviors in a distributed system.

Figure 2. Front view of the MIT AI Lab's Genghis robot.
It is a six-legged walking robot with two degrees of freedom per leg.

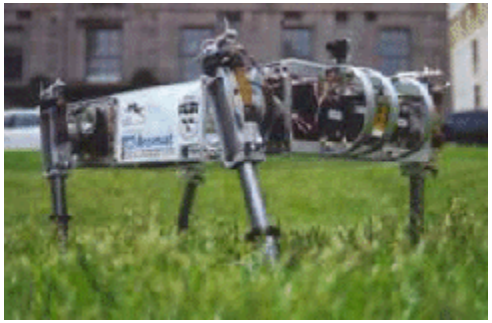
Background (Cont.)

- MIT's Leg Lab has developed several hopping, walking, and running robots.
  - The Leg Lab has been able to generalize multi-leg walking from a "virtual biped" system.
  - The Quadraped (1984 - 1987) demonstrated that two legged running algorithms could be generalized to allow four legged running.



Background (Cont.)

- McGill University built a series of small four and six legged robots with emphasis on simplicity of design and naturalness of operation.
  - Radio Control components and off the shelf components used.
  - Designs based on the belief that walking and biological is carried by simple and natural principles embedded in complex biological systems.
  - Scout II and Rhex, both capable machines.



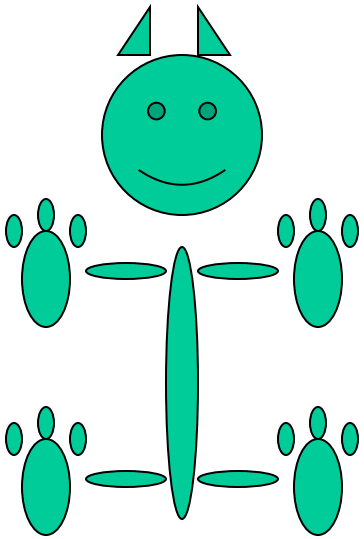


Figure 1. Standing

The Leg Lab has been able to generalize multi-leg walking into a “virtual biped” system. That is groups of legs are collected together and carry out the function of the right leg or left leg of a biped. In Figure 1, the robot has all four feet in contact with the ground, it is standing. The trotting gait, see Figure 2., for a four-legged robot pairs the front left leg with the rear right and the front right with the rear left. For galloping, see Figure 3., the front legs are a pair and the rear legs are the other pair. Figure 4., shows pacing, the left side legs are paired and the right side legs are paired.

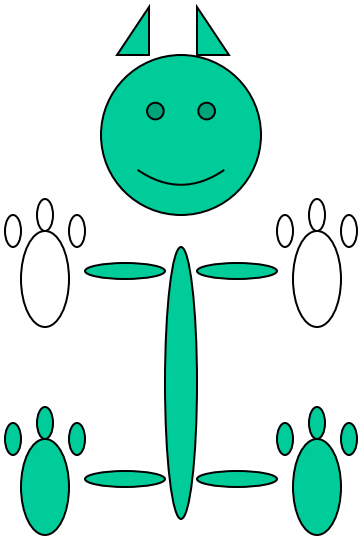


Figure 3. Galloping

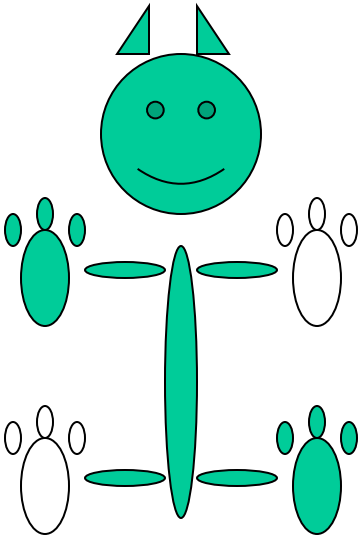


Figure 2. Trotting

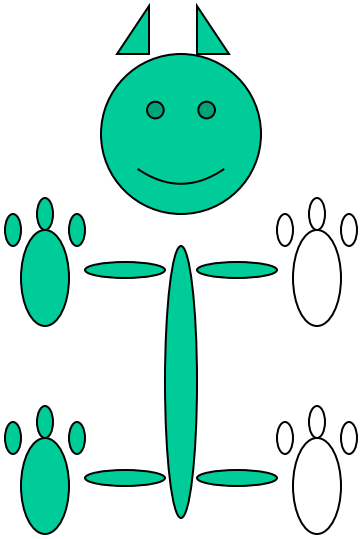


Figure 4. Pacing

Hardware and Software Design of the third prototype Stubby

- First prototype one was unpowered, it was used for developing construction and material knowledge.
- Second prototype, “Mickey” was proportioned like a house cat, but was underpowered and statically and dynamically unstable.
- Stubby is about the size and weight of a house cat but its limbs are shorter and are the in the front and back. This decreases the torque demand on the motors, and increases stability, and reduces control complexity, but looks less like a cat.

Hardware and Software(Cont.)

- The major difference between our prototypes and those of McGill and MIT is the joint structure. Our joints fold in a single plane, theirs usually are telescoping, (Quadraped, Scout II) or pantograph (Genghis).
  - The other methods are effective, but Stubby was trying to emulate a cat, or a tiger.
- Standard RC servos are used for locomotion. They are controlled by two Scott Edwards Electronics, Inc., Mini SSC II boards linked to the serial port of a laptop computer. Inputs from the foot sensors are collected by a National Instruments DAQCard-1200 card.

Hardware and Software (Cont.)

- Originally the control software was written in C.
- Recent revisions have used National Instruments LabView.
  - Provides simple GUI support.
  - Easy device interfacing for sending servo commands.
  - Easy device interfacing for collecting data from foot mounted micro-switches.
  - Multi-threading is natural for LabView.

Hardware and Software (Cont.)

- ***Input:*** popSize: int; Population size
- NumGens: int; Number of generations
- Sequence: array of frames; A frame sequence from the Movie Frame method.
- MuFact: float; The mutation factor.

- ***Output:*** newSeq array of frame; Genetically improved set of frames.
- Sequence of commands to RS232 port

- ***Function Grow***(popSize, NumGens, Sequence, MuFact, newSeq)
- {
- Population = CreatePopulation(Sequence, MuFact)
- For (1 to NumGens) {
- For (Each member of Population) {
- Move Robot to starting position.
- Execute walking sequence
- Measure and record distance traveled.
- } /* End For. */
- Rank Population based on distanced traveled.
- Select two highest ranked members of Population (Mom, Pop).
- Store Mom and Pop on Disk in case of restart.
- BreedPopulation(Mom, Pop, MuFact)
- } /* End For. */
- }/* End Grow */

Hardware and Software (Cont.)

- Resutlts? Crude but worked. Nearly doubled walking distance in 16 steps after about 200 generations.
- Although the distance improved, the ranking function did not force any smoothness, Stubby walked but walked *ugly*.
- Several more generations would be needed to get good smooth walking.

Continuing Development

- Reflexive actions.
  - Toe stubbing is one of the largest problems for Stubby. A “dodge the floor” behavior would alleviate backward pushes created by toe stubbing.
  - A “Find the floor” behavior would increase leg thrust efficiency.
  - Prototype foot with micro-switches to detect stubbing or foot down conditions has been built.
- Gait adjustment
  - Reflexive actions are the result of gait errors. When reflexive actions become frequent, a gait altering behavior should optimize the gait until frequency of the reflexive actions decreases.

Continuing Development (Cont.)

- Gait recall
  - Consider the following example.
    - A person is walking blindfolded through a parking lot and then into a field of thick tall grass. Upon the transition, the person would naturally start picking up their feet a little higher. The first few steps may be a surprise, but after that it would become routine and they would not be surprised again, until the walking surface changed again.

Continuing Development (Cont.)

- Gait Recall
  - Transition between surfaces should cause a somewhat distinct reflex generation pattern.
  - Reflex pattern is used as a key to associative memory holding the appropriate gait.
  - After new gait is being used reflexive action should drop off; if not the gait optimization process will begin using the new gait as starting point.

Walking with sensor equipped feet.

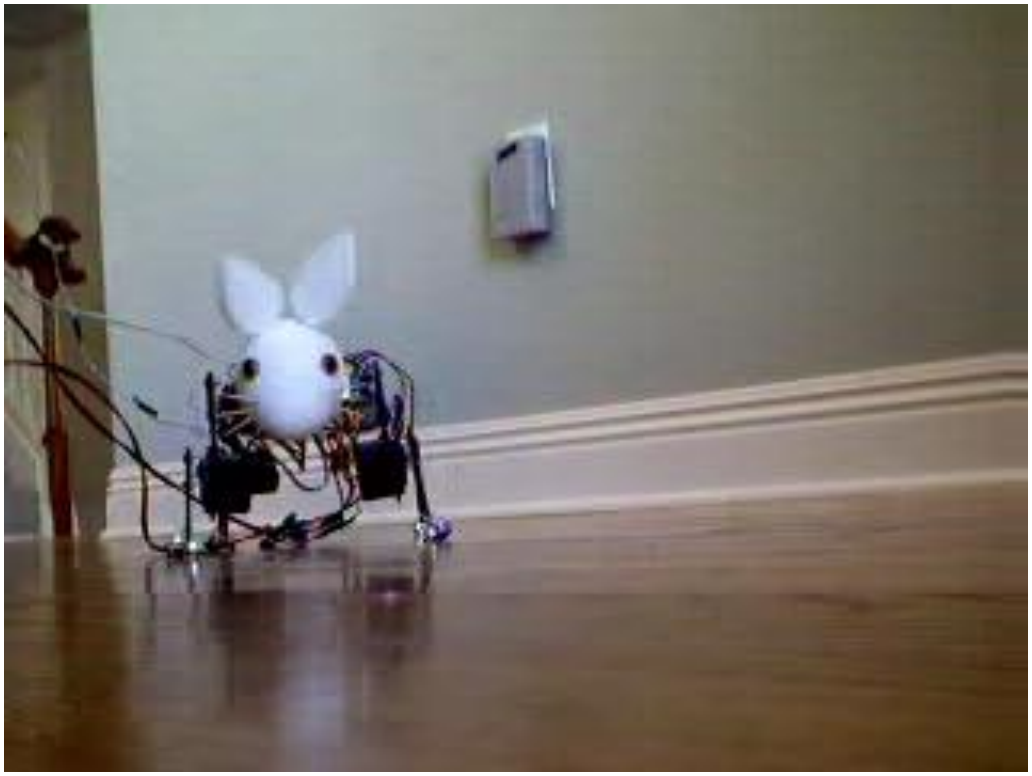


Figure 1. Stubby goes for a walk, weight of control cord slows down the robot towards the end.