

EE445M/EE360L.6 Embedded and Real-Time Systems/ Real-Time Operating Systems

Lecture 11: Robot Design, Teams, Control Systems, PID & Fuzzy Control

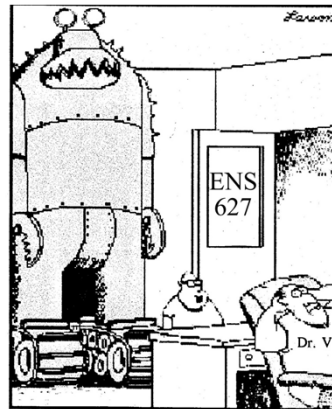
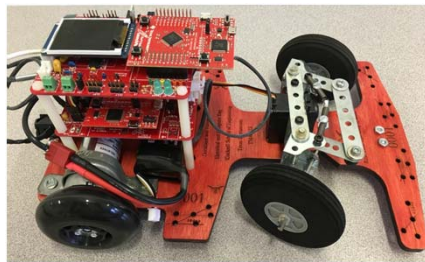
Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

1

Robot Design

- Teams
- Design process



"My project's ready for grading, Dr. Big Nose...
Hey! ... I'm talking to you, squid brain!"

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

2

What is a team?

"A team is a small number of people with complementary skills who are committed to a common purpose, performance goals, and approach for which they are mutually accountable."

(Katzenbach, J.R. & Smith, D.K. (1993). The Wisdom of Teams: Creating the High-performance Organization. Boston: Harvard Business School.)

*Decker, Philip, J. (1996) "Characteristics of an Effective Team," (Powerpoint)
http://www.cl.uh.edu/bpa/hadm/HADM_5731/ppt_presentations/29teams*

*Breslow, L. (1998). Teaching Teamwork Skills, Part 2. Teach Talk, X, 5.
<http://web.mit.edu/ilt/published/teamwork2.htm>. 13 May 2003.*

Building Blocks for Teams, (Website). Penn State University, <http://ltl.its.psu.edu/suggestions/teams/student/index.html>

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

3

Stages of Team Development

- **Forming**
 - The stage where team members are just becoming acquainted—the “honeymoon”
- **Storming**
 - Conflict begins as team members negotiate work assignments, discuss what to do
- **Norming**
 - Team members learn to work together—pride begins to develop
- **Performing**
 - Team settles down and most of the work gets done

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

4

Team Leader Role

- **Responsibilities:**
 - Calling meetings including finding a mutually agreeable time and place
 - Setting a meeting agenda (more on this later)
 - Facilitating the meeting (more later)
 - Monitoring progress against the plan
 - Identifying problem areas that need action
- **Some rules:**
 - The leader is not “the boss”
 - The team needs to agree on decisions and directions
 - Compromise is essential

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

5

Holding Effective Meetings

- Before the meeting
 - Name someone to be the facilitator
 - Create an agenda and send it to all team members
- Set a time limit for the meeting
- During the meeting, if issues emerge that are not on the agenda, the facilitator should:
 - Ask the team if this should be discussed now, or
 - Table the issues for the end of the meeting
- During the meeting:
 - Keep a list of decisions and actions items
 - Keep to the time commitment
 - Create an agenda for next meeting and agree on time and place
- After the meeting:
 - Send out a brief summary
 - List of action items
 - Those responsible for those actions

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

6

Brainstorming

- Select someone to be the recorder
- Invite everyone to give their ideas and input
- Write down all ideas without criticism or discussion
- After complete list is generated, return for discussion/analysis
- Carefully select the best approach or idea from the list

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

7

Brainstorming - Hints for Success

- Avoid being judgmental of others' ideas
- Try to look at all sides of an idea.
- Listen attentively and treat your teammates' opinions with respect
- Try to encourage the widest range of new ideas
- Everyone should participate
- Don't stop the idea session too soon
- Try to remove your ego from the discussions.
- Don't take the rejection of your ideas personally.

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

8

Group Communication

- Listen attentively and respect your teammates
- Ask questions
- Give constructive feedback:
 - Present your ideas forcefully, but keep an open mind.
 - Restate the original idea to be sure it's understood
 - Critique the idea, not the person
 - Be courteous
 - Be aware of body language and tone
- Meetings don't need to be a death march
 - Use humor effectively
 - Laugh with someone, do not laugh at someone

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

9

Team Problems (1)

- **Frustration** over the size of the project
 - Members think of an individual endeavor rather than a group endeavor
 - Break the project up into tasks
 - Engage all group members
 - Set realistic dates for each task

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

10

Team Problems (2): Conflict

- **Internal conflict** – An team member is experiencing a personal conflict that is interfering with his or her ability to perform
- **Individual conflict with another team member** - One team member is in conflict with another
- **Individual conflict with the entire team** - One team member is experiencing conflict with the entire team
- **Conflict between several team members** - The entire team is experiencing conflict with several other team members

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

11

Conflict Resolution

- Acknowledge that the conflict exists
- Gain common ground
 - Seek to understand all angles: Let each person state his or her view briefly.
 - Have neutral team members reflect on areas of agreement or disagreement.
 - Explore areas of disagreement for specific issues.
 - Have opponents suggest modifications to their points of view as well as others.
 - If consensus is blocked, ask opponents if they can accept the team's decision.
- Attack the issue, not each other
- Develop an action plan

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

12

An Effective Team Checklist

- Define a common goal for the project
- List tasks to be completed
- Assign responsibility for all tasks
- Develop a timeline and stick to it
- Develop and post a Gantt chart for the plan
- Document key decisions and actions from all team meetings.
- Send reminders when deadlines approach.
- Send confirmation when tasks are completed.
- Collectively review the project output for quality

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

13

Project Management

- Sensor
- Motor
- Mechanicals
- Network
- Power
- Control
- Debugging

*Break project into little tasks
Give yourself some milestones to show success*

Gantt Chart

ID	Task Name	Start	Finish	Duration	Jun 2004												
					20	21	22	23	24	25	26	27	28	29	30		
1	Organize Team All Team Meetings	1/21/2004	1/21/2004	1d													
2	Identify Alternative Project Topics Who: John, Sue	1/22/2004	1/23/2004	2d													
3	Call Team Meeting to Discuss Topics Who: Karen	1/23/2004	1/23/2004	2d													
4	Submit Team Topic Who: Karen	1/23/2004	1/23/2004	1d													
5	Team Topic Due	1/29/2004	1/29/2004	0d													◆

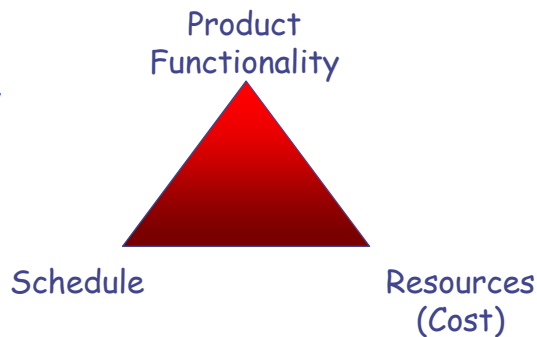
Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

14

Design Process

- We can only optimize two of the following
 - Schedule
 - Resources
 - **Functionality**



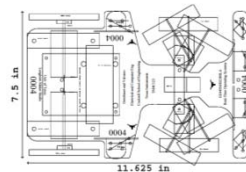
Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

15

1) Analysis Phase

- **Requirements** parameters that the system must satisfy
 - Lab 7 rules
- **Specifications** describing how the system should work
 - Robot platform
 - Tracks
 - One 11.1V battery
 - Existing motors
 - 3 minute race
- **Constraints** limitations, within the system must operate
 - The platform+\$50
 - Play nice with other robots
 - Interfaces with other instruments and test equipment
 - Development schedule



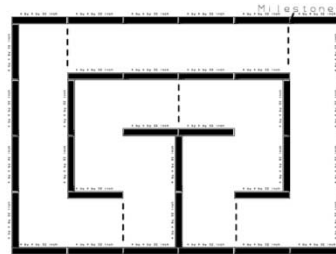
Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

16

2) High-Level Design Phase

- Project proposal
 - Build conceptual models
 - data flow graph
 - block diagrams
 - fundamental equations
 - Exploit abstraction
 - Search for existing components
 - Try different control algorithms

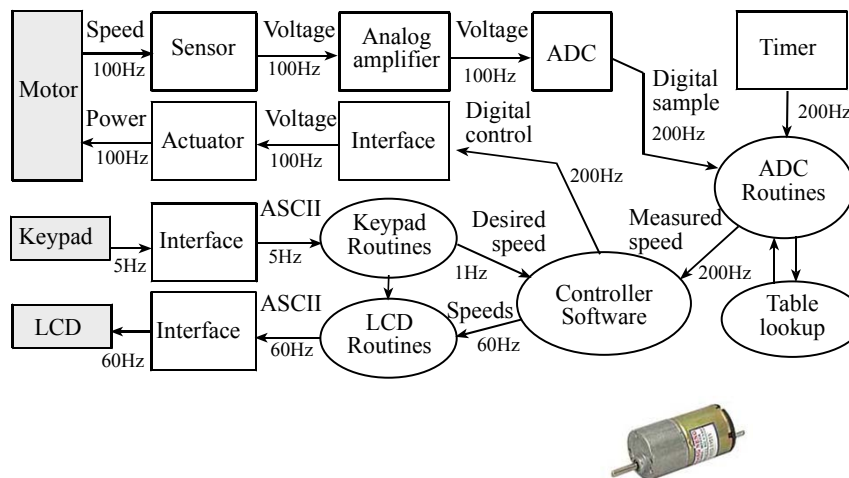


Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

17

Data Flow Graph



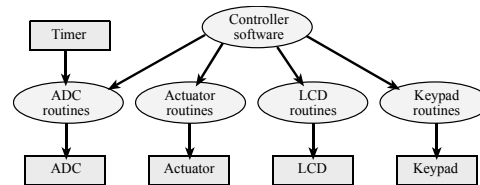
Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

18

3) Engineering Design Phase

- Hierarchical structure
 - Call-graphs
 - Data structures
 - Flow charts
- Basic I/O interfaces
- Overall software scheme
- Direct correlation between hardware/software systems and conceptual models
- Built mock-ups of the mechanical parts (connectors, chassis, cables etc.)
- Mock-ups of user software interface

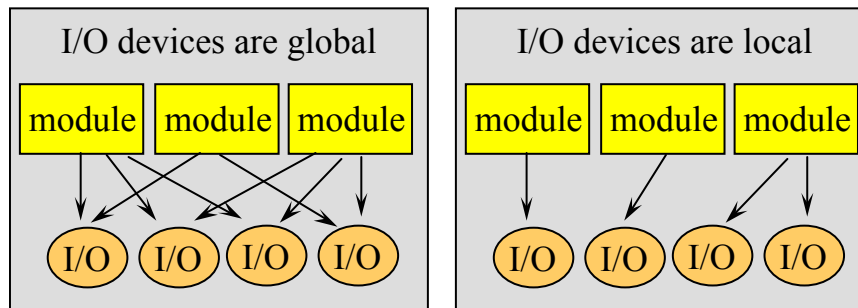


Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

19

Call Graph



Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

20

4) Implementation Phase

- Concurrent implementation
- Initially implement using simulation
- Divide into modules
- Unit-level testing and debugging

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

21

5) Testing Phase

- Design for test
- Concurrent testing
 - Bottom up, from unit-level to subsystem and system integration
- Control and observability
 - Use LCD, SDC, UART, ...

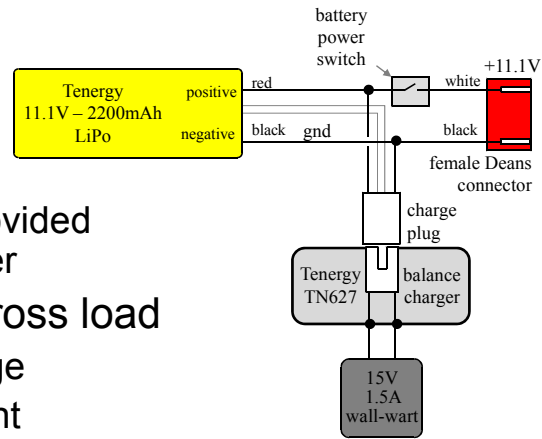
Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

22

LiPo Battery

- No built-in protection circuit module (PCM)
- Discharging
 - Avoid over-discharge
- Charging
 - Always use provided balance charger
- Test battery across load
 - Measure voltage
 - Measure current



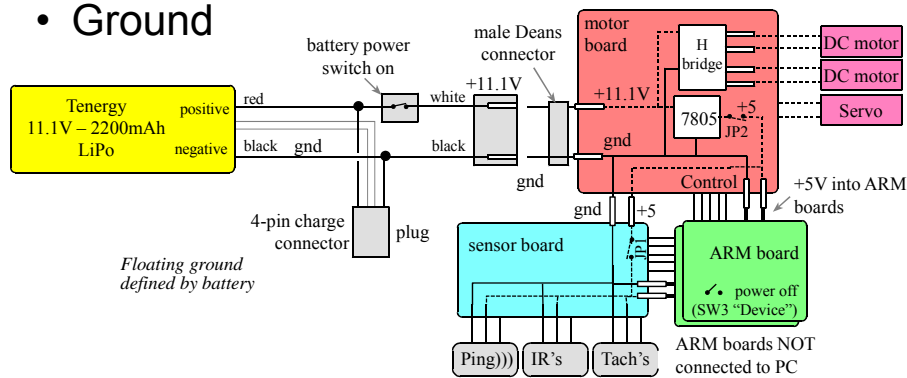
Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

23

Under Battery Power

- Current path battery-motor-battery
- One +5V
- Ground



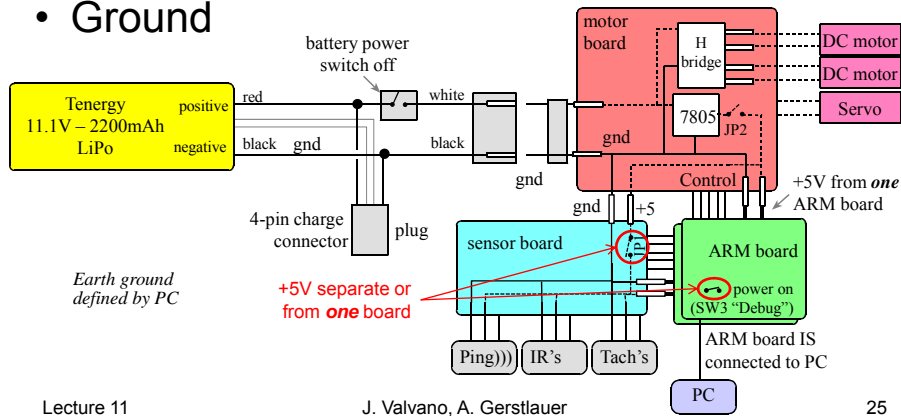
Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

24

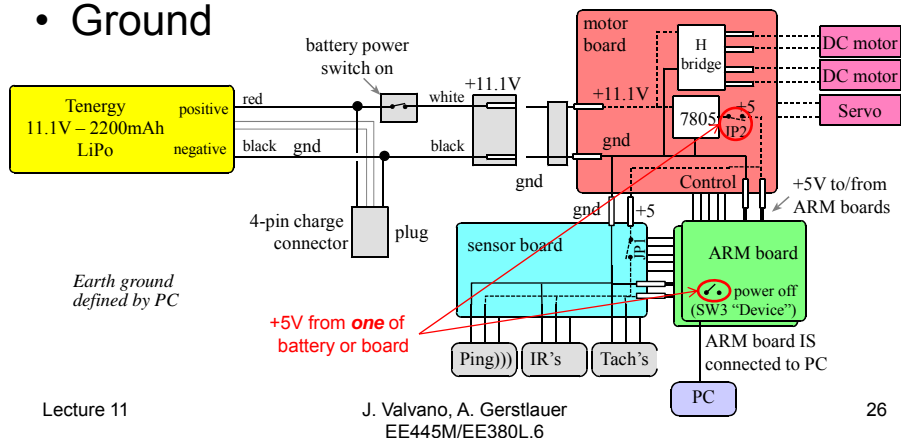
While Debugging with PC

- Battery disconnected (switch off)
- One or two separate +5V (jumper)
- Ground



Debugging with Motor

- Battery connected (switch on)
- One, two or three +5V (jumpers)
- Ground



Race Tracks

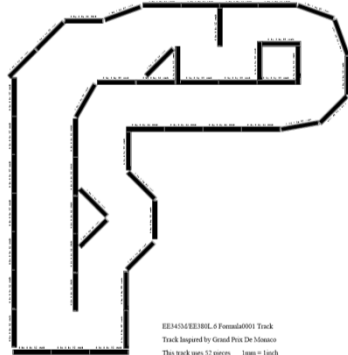


Lecture 11

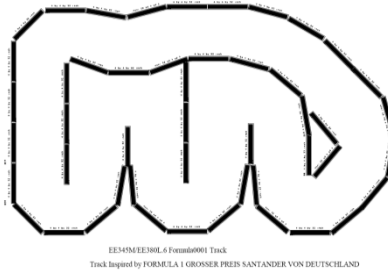
J. Valvano, A. Gerstlauer
EE445M/EE380L.6

27

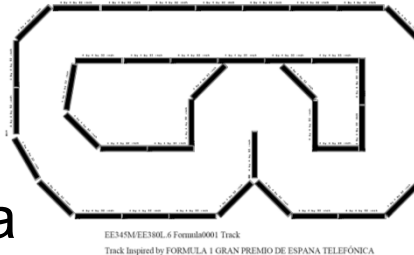
Monaco



Von Deutschland



Espana

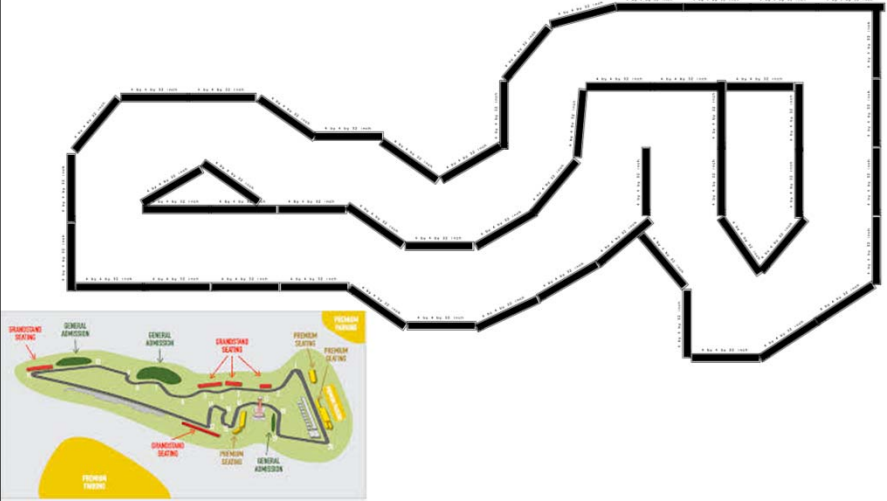


Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

28

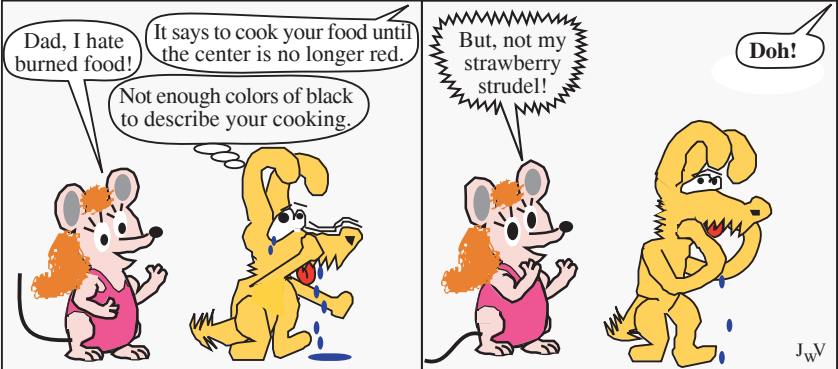
Circuit of the Americas



Lecture 11 J. Valvano, A. Gerstlauer 29
EE445M/EE380L.6

Control Systems

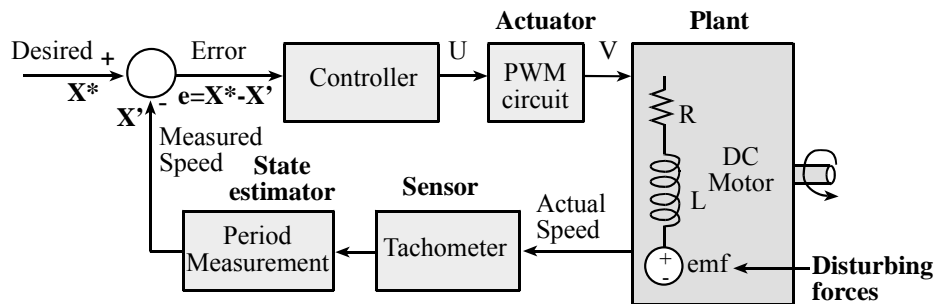
- **Easy: Incremental**
- **Linear: P, PI, PID**
- **Intuitive: Fuzzy Logic**



Lecture 11 J. Valvano, A. Gerstlauer 30
EE445M/EE380L.6

What to Control?

- Motor speed
 - Error = Desired speed – Actual speed
 - Error = Desired period – Actual period



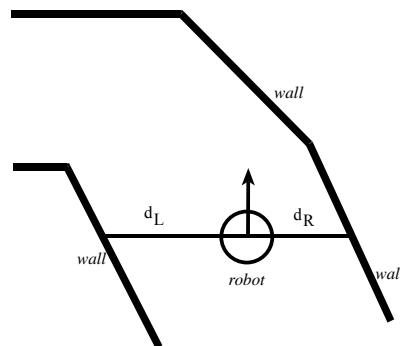
Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

31

What to Control?

- Distance to one wall
 - Error = Desired distance – Actual distance
- Equal distances to both walls
 - Error = Left distance – Right distance



Sensors

- Time constant $\rightarrow f_s$
- Filters
 - Analog, digital
- Signal/noise ratio
- Calibration accuracy, drift

Lecture 11

32

What to Control?

- Angle to most open track
 - Error = Desired – Actual angle

Turn so most open in front
Avoid making U-turns

Lecture 11 J. Valvano, A. Gerstlauer
EE445M/EE380L.6 33

What to Control?

- Angle to wall

speed is determined by distance, d, to object in front

a, b are measured distances from center of robot to the wall

Track the right wall if $(a_R + b_R) < (a_L + b_L)$
 Want to make $A_R = 90$
 If $A_R = 90$, then $b_R/a_R = \cos C_R$
 $x_R = 100 * \cos C_R$ is a calibration constant
 error = $(100 * b_R) / a_R - x_R$
 if error > 0, $A_R < 90$, turn left
 if error < 0, $A_R > 90$, turn right
 if $\min(a_R, b_R) < 20$, move left-right

Track the left wall if $(a_R + b_R) > (a_L + b_L)$
 Want to make $A_L = 90$
 If $A_L = 90$, then $b_L/a_L = \cos C_L$
 $x_L = 100 * \cos C_L$ is a calibration constant
 error = $(100 * b_L) / a_L - x_L$
 if error > 0, $A_L < 90$, turn right
 if error < 0, $A_L > 90$, turn left
 if $\min(a_L, b_L) < 20$, move right-left

Lecture 11 J. Valvano, A. Gerstlauer
EE445M/EE380L.6 34

Performance Measures

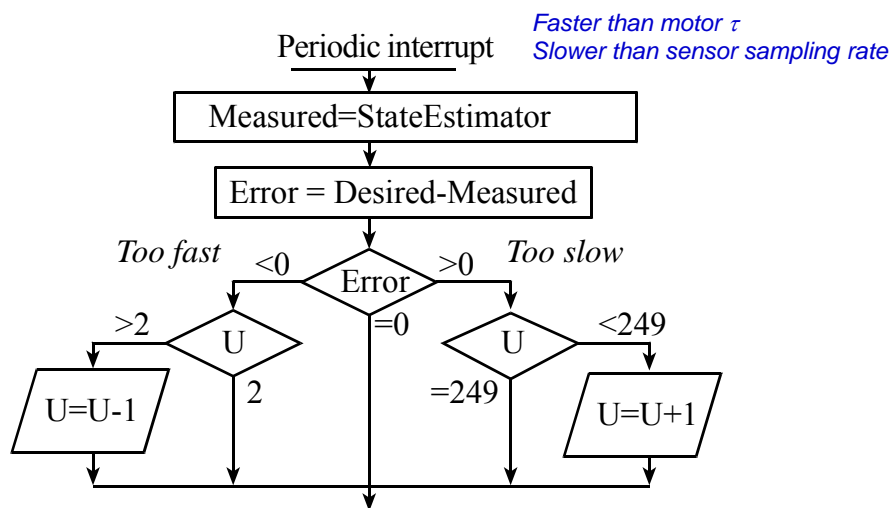
- Accuracy
 - Magnitude of the Error = Desired – Actual
- Stability
 - No oscillations
- Overshoot (underdamped, overdamped)
 - Ringing, slow
- Response Time to new steady state after
 - Change in desired setpoint
 - Change in load

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

35

Incremental Controller



Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

36

Incremental Controller

```

long Xstar; // desired
long X;     // actual, measured input
long U;     // PWM actuator output
void Timer0A_Handler(void){ long E;
    E = Xstar-X;           // error
    if(E < 0)              U--; // too fast
    else if(E > 1) U++;    // too slow
                          // close enough
    if(U < 2) U=2;        // Constrain output
    if(U > 249) U=249;
    PWM0_Duty(U);         // output
    TIMER0_ICR_R = TIMER_ICR_CAECINT;
}

```

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

37

PID Controller

$$U(t) = K_p E(t) + \int_0^t K_i E(\tau) d\tau + K_d \frac{dE(t)}{dt}$$

- Proportional $U_p = K_p E$
- Integral $U_i = U_i + K_i E \Delta t$
- Derivative $U_d = K_d (E(n) - E(n-1)) / \Delta t$
- PID $U = U_p + U_i + U_d$
- Run ten times faster than motor τ
- Run slower or equal to sensor sampling rate

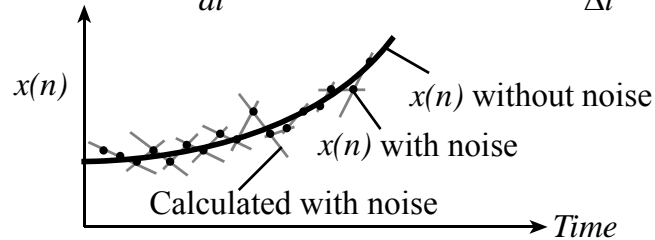
Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

38

Derivative Term

$$D(t) = K_d \frac{dE(t)}{dt} \Rightarrow D(n) = K_d \frac{E(n) - E(n-1)}{\Delta t}$$



$$D(n) = K_d \left(\frac{1}{2} \frac{E(n) - E(n-3)}{3\Delta t} + \frac{1}{2} \frac{E(n-1) - E(n-2)}{\Delta t} \right) \quad \begin{array}{l} \bullet \text{Analog filter} \\ \bullet \text{Digital filter} \end{array}$$

$$D(n) = K_d \left(\frac{E(n) + 3E(n-1) - 3E(n-2) - E(n-3)}{6\Delta t} \right) \quad \bullet \text{Linear regression}$$

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

39

PI Controller

```
void Timer0A_Handler(void){
    E = Xstar-X;
    P = (105*E)/20;           // Kp = 105/20
    I = I+(101*E)/640;       // KiΔt = 101/640
    if(I < -500) I=-500;    // anti-reset windup
    if(I > 4000) I=4000;
    U = P+I;                  // PI controller
    if(U < 100) U=100;       // Constrain output
    if(U>19900) U=19900;
    PWM0_Duty(U);           // output
    TIMER0_ICR_R = TIMER_ICR_CAECINT;
}
```

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

40

Laplace Domain

$$G(s) = K_p + K_d s + \frac{K_i}{s}$$

$$H(s) = \frac{m}{1 + \tau s}$$

$$\frac{X(s)}{X^*(s)} = \frac{G(s)H(s)}{1 + G(s)H(s)}$$

Lecture 11 J. Valvano, A. Gerstlauer 41
EE445M/EE380L.6

Motor Parameters

- Invoke a step, measure response
 - Time lag, time constant, τ
- Plot speed in RPM versus duty cycle
 - Sensitivity m

Real \neq Theory

$$H(s) = \frac{m}{1 + \tau s}$$

Lecture 11 J. Valvano, A. Gerstlauer 42
EE445M/EE380L.6

Controller Tuning

- Start with just a proportional term (**K_p**)
 - proportional controller will generate a smooth motor speed
 - choose the sign of the term K_p so the system is stable
 - try different K_p constants until the response times are fast enough
- The next step is to add some integral term (**K_i**)
 - a little at a time
 - to improve the steady state controller accuracy
 - without adversely affecting the response time
 - choose the sign of the term K_i so the system is stable
 - Don't change both K_p and K_i at once.
- The last step is the derivative term (**K_d**)
 - a little at a time
 - reduce the overshoots/undershoots in the step response
 - choose the sign of the term K_d so the overshoots/undershoots are reduced

Highly nonlinear -> empirical approach

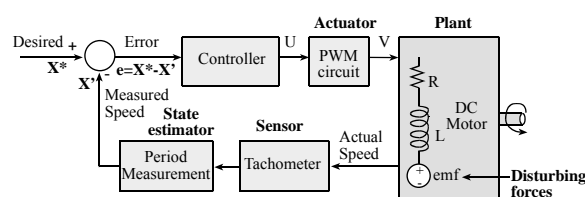
Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

43

Fuzzy Logic Controller

- Incremental
 - + Simple, stable
 - Slow response
- PID or PI
 - + Theory, fast response
 - Needs empirical tuning, depends on load
- Fuzzy Logic *Maps human intuition into rules*
 - + Fast, good when you have expert knowledge
 - + Abstractive approach
 - Needs empirical tuning



Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

44

Fuzzy Membership Set

- Membership set, variable, set
 - Value specifying levels of truth
 - Collection describes the entire system

0.....32.....64.....96.....128.....160.....192.....224.....255

Not at all...a little bit...somewhat...mostly...pretty much...definitely

- Examples for a speed control system
 - TooSlow
 - SpeedOK
 - TooFast
 - SlowingDown
 - SpeedConstant
 - SpeedingUp

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

45

Fuzzy Membership Set

- Lab 7 example membership sets
 - Too close to the right wall
 - Distance to the right wall is ok
 - Too far away from the right wall
 - Too close to the left wall
 - Distance to the left wall is ok
 - Too far away from the left wall
 - Open space to 30 degrees to the right
 - Open space to straight ahead
 - Open space to 30 degrees to the left

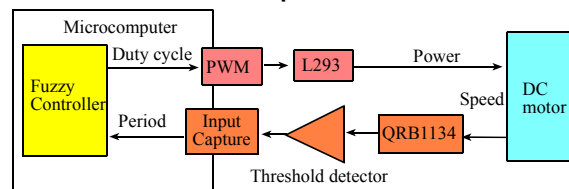
Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

46

Speed Controller

- Desired state
 - X^* is the desired tach period
- Physical plant
 - X real state variable, actual period
- State estimator, data acquisition
 - X' measured tach period



Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

47

Fuzzy Approach

- Preprocessor
 - Crisp inputs (variables with units)
- Fuzzification
 - Input membership sets
- Fuzzy rules
 - Output membership sets
- Defuzzification
 - Crisp outputs (variables with units)
- Postprocessor and actuator output

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

48

Preprocessor

- Crisp inputs
 - $E = X^* - X'$ error in motor period
 - $D = X'(n) - X'(n-1)$ acceleration

```

unsigned char Ts;      // Desired Speed in 3.9 rpm units
unsigned char T;      // Current Speed in 3.9 rpm units
unsigned char Told;   // Previous Speed in 3.9 rpm units
char D;               // Change in Speed in 3.9 rpm/time units
char E;               // Error in Speed in 3.9 rpm units
void CrispInput(void){
    E=Subtract(Ts,T);
    D=Subtract(T,Told);
    Told=T;           /* Set up Told for next time */
}

```

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

49

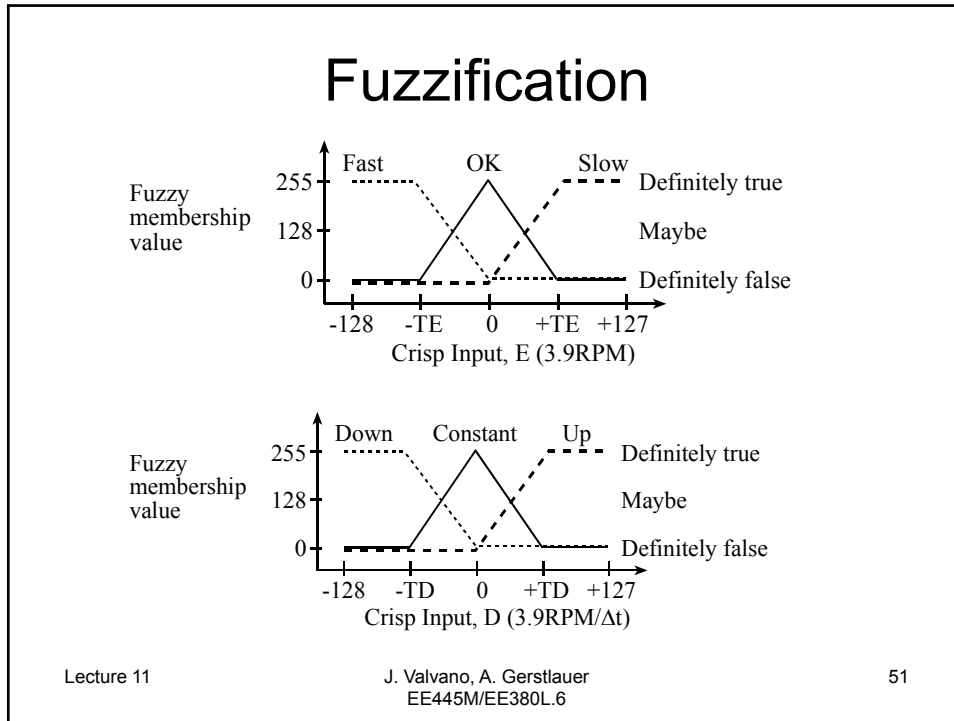
Fuzzification

- Preprocessor, crisp inputs
 - $E = X^* - X'$ error in motor period
 - $D = X'(n) - X'(n-1)$ acceleration
- Fuzzification
 - Slow* True if the motor is spinning too slow
 - OK* True if the motor is spinning at the proper speed
 - Fast* True if the motor is spinning too fast
 - Up* True if the motor speed is getting larger
 - Constant* True if the motor speed is remaining the same
 - Down* True if the motor speed is getting smaller.

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

50



Fuzzification

```

#define TE ???
long Fast, OK, Slow, Down, Constant, Up;
#define TD ???
long Increase, Same, Decrease;
#define TN ???
void InputMembership(void){
    if(E <= -TE) { /* E <= -TE */
        Slow = 255;
        OK = 0;
        Fast = 0;}
    else
        if (E < 0) { /* -TE<E<0 */
            Slow = (255*(-E))/TE;
            OK = 255-Slow;
            Fast = 0;}
        else
            if (E < TE) { /* 0<E<TE */
                Slow = 0;
                Fast = (255*E)/TE;
                OK = 255-Fast;}
            else { /* +TE <= E */
                Slow = 0;
                OK = 0;
                Fast = 255;
            }
    }

    if(D <= -TD) { /* D <= -TD */
        Up = 255;
        Constant = 0;
        Down = 0;}
    else
        if (D < 0) { /* -TD<D<0 */
            Up = (255*(-D))/TD;
            Constant = 255-Up;
            Down = 0;}
        else
            if (D < TD) { /* 0<D<TD */
                Up = 0;
                Down = (255*D)/TD;
                Constant = 255- Down;}
            else { /* +TD <= D */
                Up = 0;
                Constant = 0;
                Down = 255;
            }
    }
}
    
```

Lecture 11 J. Valvano, A. Gerstlauer 52
EE445M/EE380L.6

Fuzzy Rules

E \ D	Down	Constant	Up
Slow	Increase	Increase	
OK	Increase	Same	Decrease
Fast		Decrease	Decrease

If *OK* and *Constant* then *Same*
 If *OK* and *Up* then *Decrease*
 If *Fast* and *Constant* then *Decrease*
 If *Fast* and *Up* then *Decrease*
 If *OK* and *Down* then *Increase*
 If *Slow* and *Constant* then *Increase*
 If *Slow* and *Down* then *Increase*

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

53

Fuzzy Rules

Same=(*OK* and *Constant*)

Decrease=(*OK* and *Up*) or (*Fast* and *Constant*) or (*Fast* and *Up*)

Increase=(*OK* and *Down*) or (*Slow* and *Constant*) or (*Slow* and *Down*)

- **and** operation is minimum

```

unsigned char static min(unsigned char u1,unsigned char u2){
    if(u1>u2) return(u2);
    else return(u1);
}

```

- **or** operation is the maximum

```

unsigned char static max(unsigned char u1,unsigned char u2){
    if(u1<u2) return(u2);
    else return(u1);
}

```

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

54

Fuzzy Rules

	D	Down	Constant	Up
E				
Slow		Increase	Increase	
OK		Increase	Same	Decrease
Fast			Decrease	Decrease

```

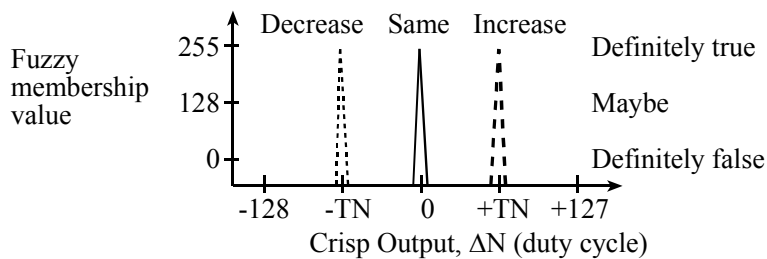
void OutputMembership(void){
    Same=min(OK,Constant);
    Decrease=min(OK,Up)
    Decrease=max(Decrease,min(Fast,Constant));
    Decrease=max(Decrease,min(Fast,Up));
    Increase=min(OK,Down)
    Increase=max(Increase,min(Slow,Constant));
    Increase=max(Increase,min(Slow,Down));
}
    
```

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

55

Defuzzification



$$\Delta N = (Decrease \cdot (-TN) + Same \cdot 0 + Increase \cdot TN) / (Decrease + Same + Increase)$$

```

long dN;
void CrispOutput(void){
    dN= ( TN* ( Increase-Decrease ) ) / ( Decrease+Same+Increase );
}
    
```

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

56

Fuzzy Logic Controller

```

void Timer0A_Handler(void){
  T = SE();           // estimate speed, set T, 0 to 255
  CrispInput();       // Calculate E,D and new Told
  InputMembership();  // Sets Fast,OK,Slow,Down,Constant,Up
  OutputMembership(); // Sets Increase,Same,Decrease
  CrispOutput();      // Sets dN
  N = max(0,min(N+dN,255));
  PWM0_Duty(N);      // output to actuator, Program 8.4
  TIMER0_ICR_R = TIMER_ICR_CAECINT;// ack
}

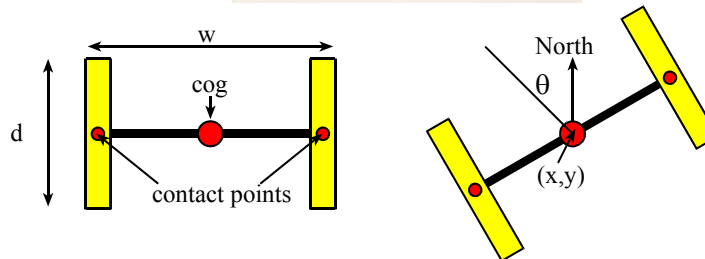
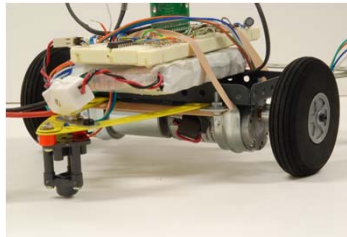
```

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

57

Odometry



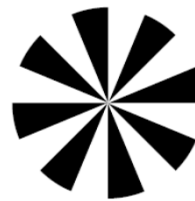
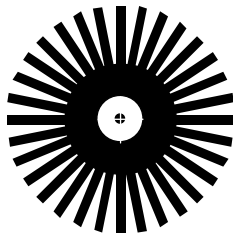
Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

58

Constants

- Number of slots/rotation, $n=32$
- Wheel diameter, $d=886$ (0.01cm)
- Wheelbase, $w=1651$ (0.01cm)
- Wheel circumference, $c=\pi d=2783$ (0.01 cm)



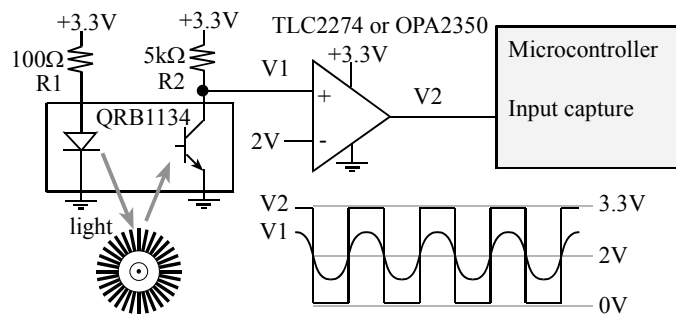
Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

59

Measurements

- LCount the number of left slots in Δt
- RCount the number of right slots in Δt
- Counts vary from -28 to +28 each Δt



Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

60

Simple Cases

- $-28 \leq m \leq +28$ each Δt

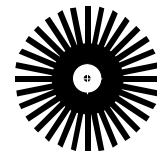
<i>LCount</i>	<i>RCount</i>	<i>Motion</i>
m	m	straight line motion in the current direction
0	m	pivot about stopped left motor
m	0	pivot about stopped right motor
m	-m	pure rotation about cog

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

61

Derivations



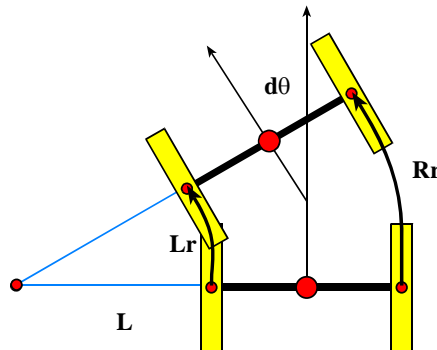
c = circumference

n = number of slots per rotation

w = wheelbase

$L_r = LCount * c / n$ the arc distance traveled by the left wheel (0.01cm)

$R_r = RCount * c / n$ the arc distance traveled by the right wheel (0.01cm)



Assume L_r, R_r positive
Assume $R_r > L_r$

$$\begin{aligned} L/L_r &= (L+w)/R_r \\ L/L_r - L/R_r &= w/R_r \\ L R_r - L L_r &= w L_r \\ L &= w L_r / (R_r - L_r) \end{aligned}$$

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

62

Derivations

$d\theta = Lr/L = Rr/(L+w)$

Lecture 11 J. Valvano, A. Gerstlauer 63
 EE445M/EE380L.6

Derivations

We can divide the change in position into two components

$dz = (L+w/2) \cdot \tan(d\theta/2)$

if $d\theta$ is small

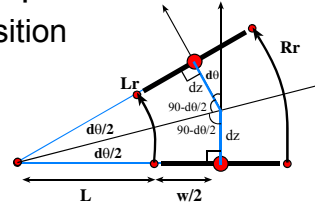
$dz = d\theta/2 \cdot (L+w/2)$

Lecture 11 J. Valvano, A. Gerstlauer 64
 EE445M/EE380L.6

Odometry

- Needs very accurate sensors
 - Errors accumulate
- OK for relative travel from known position
 - periodic absolute knowledge of position

$L_r = L_{\text{Count}} * c / n$	(0.01cm)
$R_r = R_{\text{Count}} * c / n$	(0.01cm)
$L = (w * L_r) / (R_r - L_r)$	(0.01cm)
$d\theta = (100 * L_r) / L$	(0.01 radians)
$dz = ((d\theta/2) * (L + w/2)) / 100$	(0.01cm) <i>approximation</i>
$x = x + dz * \cos(\theta)$	(0.01cm)
$y = y + dz * \sin(\theta)$	(0.01cm) <i>first part of move</i>
$\theta = \theta + d\theta$	(0.01 radians)
$x = x + dz * \cos(\theta)$	(0.01cm)
$y = y + dz * \sin(\theta)$	(0.01cm) <i>second part of move</i>



Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

65

Things that can go bad

- Hitting the wall
 - Think of three ways to tell if you hit the wall
 - Corrective measures
- Wrong-way Dayo
 - Think of ways to reduce the chances
 - Three repairs -> disqualification
- Other robots in the way
 - Can you distinguish a robot from a wall?
 - Strategy for passing

Lecture 11

J. Valvano, A. Gerstlauer
EE445M/EE380L.6

66