

Introdução à Robótica

Sensores (Tratamento do sinal)

Prof. Douglas G. Macharet
douglas.macharet@dcc.ufmg.br

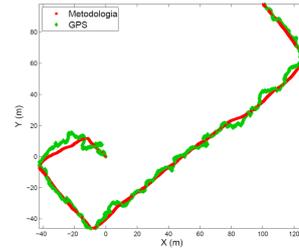
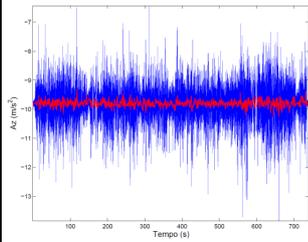
*Apresentação baseada nos slides de Dra. Linda Bushnell.



Introdução

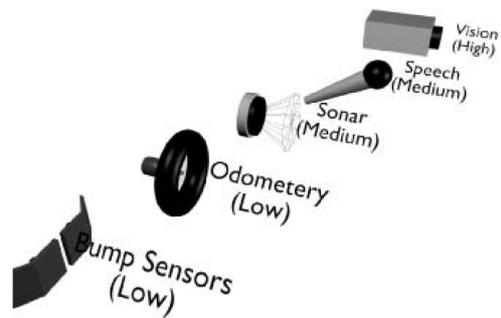
- Grande problema dos sensores é o ruído
 - Dificulta a interpretação dos dados
- Sensores com diferentes características
- Como separar a informação útil?

Introdução



Introdução

Diferentes níveis de processamento



Introdução

- Abordagens para o problema
 - Filtragem
 - Calibração



Filtragem

- Aplicações básicas de filtros
 - Separação de sinais combinados
 - Restauração de sinal distorcido
- Filtros
 - Analógicos
 - Circuitos (resistores, indutores e capacitores)
 - Digitais
 - Processador digital (cálculos numéricos)



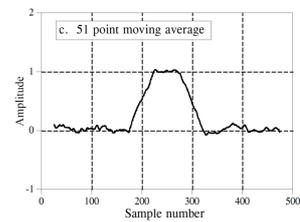
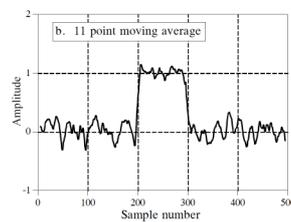
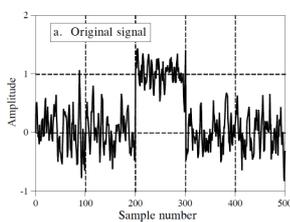
Filtragem

- Média móvel
- Passa-banda
 - Alta
 - Baixa
- Faixa
 - Passa / Rejeita



Filtragem

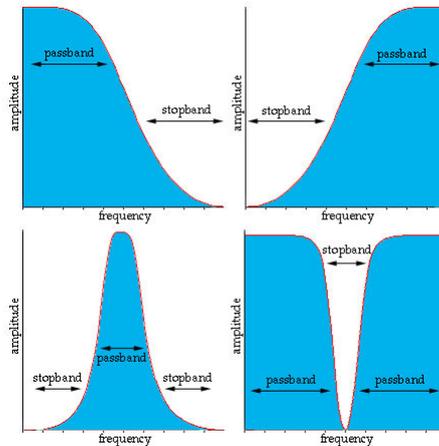
Média móvel



$$y[i] = \frac{1}{M} \sum_{j=0}^{M-1} x[i+j]$$



Filtragem



Filtragem

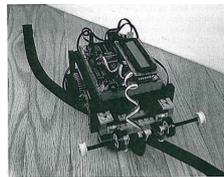
- Problema na interpretação dos dados
 - Dificuldade de se tomar uma decisão
 - Comportamento diferente do desejado
- Tarefa de exemplo
 - Seguir uma faixa de outra cor na superfície
 - Utilização de um sensor óptico
 - Ruído, influência do meio, ...



Filtragem

Tarefa exemplo (*line-following*)

- Um sensor OR apontado para superfície
- Movimentação
 - Seguir para frente (giro)
 - Ao cruzar a linha inverter a direção



Filtragem

Tarefa exemplo (*line-following*)

```
void line_follow() {
    while (1) {
        waddle_left();
        waituntil_on_the_line();
        waituntil_off_the_line();
        waddle_right();
        waituntil_on_the_line();
        waituntil_off_the_line();
    }
}
```

```
void waddle_left() {
    fd(RIGHT_MOTOR);
    off(LEFT_MOTOR);
}

void waddle_right() {
    fd(LEFT_MOTOR);
    off(RIGHT_MOTOR);
}
```

Filtragem

Tarefa exemplo (*line-following*)

- Como implementar as seguintes funções?

```
waituntil_on_the_line();  
waituntil_off_the_line();
```

- Como saber se está sobre a linha (ou fora)?
 - Utilizar um valor de referência
 - O ruído irá influenciar esse valor?



Filtragem

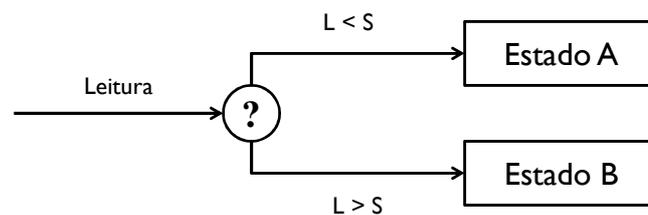
- Limiar (*thresholding*)
 - Fixo
 - Parametrizado
 - *Hysteresis*
 - Adaptativo



Filtragem

Thresholding fixo

- Maneira mais simples de tratar os dados
- Leitura comparada com um *setpoint* fixo



Filtragem

Thresholding fixo

- Ideia semelhante a um quantizador
 - Converter um valor contínuo da leitura do sensor para um estado digital (binário)
- Tarefa exemplo
 - Leitura na superfície ≈ 10
 - Leitura na linha preta ≈ 50
 - Qual valor utilizar como *setpoint*?



Filtragem

Thresholding fixo (line-following)

```
void waituntil_on_the_line() {  
    while(line_sensor() < 30);  
}  
  
void waituntil_off_the_line() {  
    while(line_sensor() > 30);  
}  
  
int line_sensor() {  
    return analog(0);  
}
```



Filtragem

Thresholding fixo

- Leituras sofrem influência do meio
 - E se esse valor mudar com as condições atuais?
 - Procurar todos os locais do código e alterar?
 - Como resolver?
- *Thresholding* fixo parametrizado
 - Centralização na definição do valor
 - Pode ser informado pelo usuário (calibração)



Filtragem

Thresholding fixo parametrizado (*line-following*)

```
#define LINE_SETPOINT= 30;

void waituntil_on_the_line() {
    while (line_sensor() < LINE_SETPOINT);
}

void waituntil_off_the_line() {
    while (line_sensor() > LINE_SETPOINT);
}
```



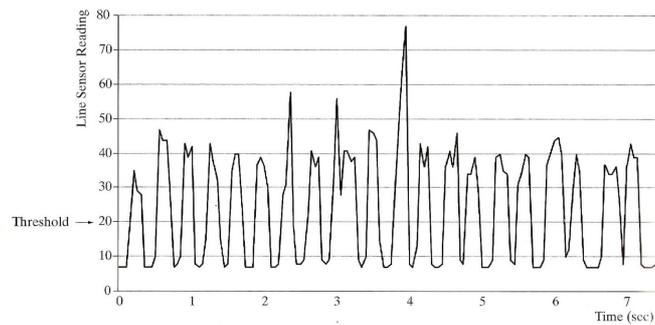
Filtragem

- Qual o problema com *thresholding* fixo?
 - As leituras não são muito confiáveis
 - Ruído, influência do meio (*glitches*), ...
 - É difícil escolher o melhor (único) *setpoint*



Filtragem

Tarefa exemplo (*line-following*)



Filtragem

Thresholding com Hysteresis

- Utilizar dois *setpoints* para reduzir os erros
- *Hysteresis*
 - O estado anterior do sistema (sobre/fora) afeta a decisão do sistema de ir para um novo estado
- Tarefa exemplo
 - Esperar até uma leitura mais forte da superfície
 - Esperar até uma leitura mais forte da linha



Filtragem

Thresholding com Hysteresis (line-following)

```
int LINE_SETPOINT = 35;
int FLOOR_SETPOINT = 10;

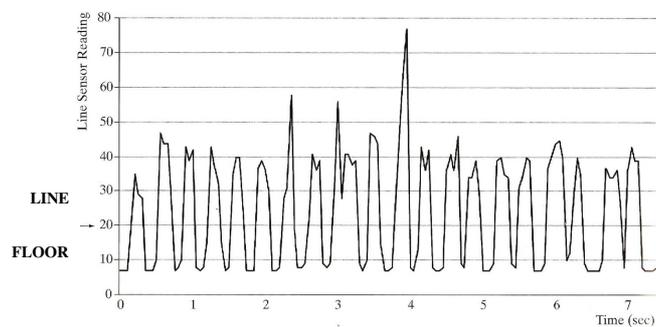
void waituntil_on_the_line() {
    while (line_sensor() < LINE_SETPOINT);
}

void waituntil_off_the_line() {
    while (line_sensor() > FLOOR_SETPOINT);
}
```



Filtragem

Tarefa exemplo (line-following)



Filtragem

Thresholding com Hysteresis

- Evita tomar decisões precipitadas
 - Ponto mais escuro na superfície
 - Ponto mais claro sobre a linha preta
- *Hysteresis*
 - Robustez
 - Ajuda na interpretação dos dados



Filtragem

Adaptativo

- O *setpoint* é modificado durante a execução
 - Adaptação de acordo com a variação do sinal
 - Requer um entendimento do fenómeno
 - Possíveis valores e características do sensor
 - Ex: Utilizar o valor médio das leituras



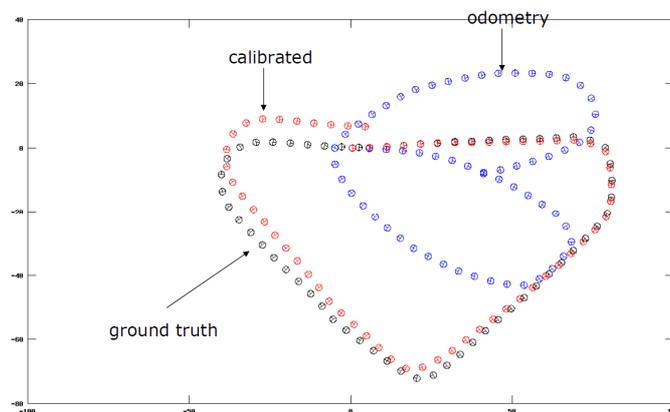
Calibração

- Identificar características do sinal
 - Comparação entre medidas
 - Adaptar o programa a diferentes situações
 - Ex: Diferentes condições de iluminação
- Maneira básica de modelar o sensor
- Geralmente realizada antes da tarefa



Calibração

Exemplo – Odometria



Calibração

- Calibração por demonstração
 - Ex: Alterar os *setpoints* sem recompilar o código
- Considerar o contexto atual do sistema
 - Melhor adaptação a diferentes condições
 - Não é necessário um conhecimento prévio
- Tarefa exemplo
 - Posicionar o robô sobre a superfície/linha



Calibração

Tarefa exemplo (*line-following*)

```
int LINE_SETPOINT= 100;
int FLOOR_SETPOINT= 100;

void main() {
    calibrate();
    line_follow();
}
```



Calibração

Tarefa exemplo (*line-following*)

```
void calibrate() {
  int new;
  while (!start_button()) {
    new = line_sensor();
    printf("Line: old=%d new=%d\n", LINE_SETPOINT, new);
    msleep(50L);
  }
  LINE_SETPOINT= new; /* accept new value */
  beep(); while (start_button()); /* debounce button press */

  while (!start_button()) {
    new = line_sensor();
    printf("Floor: old=%d new=%d\n", FLOOR_SETPOINT, new);
    msleep(50L);
  }
  FLOOR_SETPOINT= new; /* accept new value */
  beep(); while (start_button()); /* debounce button press */
}
```



Calibração

- Calibração persistente
 - Evitar a necessidade de recalibrar ao ligar
 - Reduzir o trabalho envolvido na inicialização

```
persistent int setpoint; /* persistente global , ? */
```

- Não são inicializadas (valor inicial arbitrário)



Calibração

Histórico do sensor (*sensor histories*)

- Evitar a necessidade de calibração manual
 - Determinar automaticamente os *thresholds*
 - Ajustar dinamicamente a diferentes condições
 - Robustez
- Como fazer isso?
 - Ex: Manter um histórico dos valores mínimo e máximo, calcular o *threshold* durante a execução



Calibração

Histórico do sensor (*line-following*)

```
int line, LINE_THRESHOLD;
int max=0;
int min=255;

int line_sensor() {
    line = analog(0);
    if (line > max) max = line;
    if (line < min) min = line;
    LINE_THRESHOLD = (max + min) / 2;
    return line;
}
```



Calibração

Histórico do sensor (*sensor histories*)

- Problema
 - Utilizar apenas o máximo/mínimo não produzirá bons resultados dependendo da situação
 - Ex: O *threshold* pode ficar acima do real (ruído)
- Solução
 - Considerar todo o histórico de leituras
 - Calcular o valor da leitura média



Calibração

Histórico do sensor (*sensor histories*)

- Histórico das leituras dos sensores (rotinas)
- Interactive C (*senshist.lis*)
 - *senshist.icb*
 - *senshist.c*
 - *sensproc.icb*

