Construção de um pêndulo invertido sobre um robô móvel controlado com o executivo SHaRK

Marco Leonor, Márcio Neves

***Resumo* – Este artigo descreve a construção do sistema de controlo de um pêndulo invertido, equilibrado sobre uma base móvel semelhante às utilizadas por pequenos robôs móveis. O controlo é efectuado num PC de apoio recorrendo ao executivo SHaRK. A comunicação entre a base móvel e o PC é efectuada pela porta série, recorrendo a pequeno adaptador também realizado no âmbito deste trabalho. É ainda de referir que, para a definição do tipo de controlador a utilizar, se fez modelizou o sistema recorrendo ao Matlab. O trabalho foi efectuado no âmbito da disciplina de Sistemas de Tempo-Real do 5º ano da LEET e LECT.**

I. INTRODUÇÃO

O trabalho descrito neste artigo enquadra-se na componente prática da disciplina de Sistemas de Tempo-Real da LEET/LECT [1] e consta da construção de um pêndulo invertido sobre uma plataforma móvel, com controlo efectuado por um PC de apoio munido do executivo (kernel) de tempo real SHaRK [2]. Este kernel, desenvolvido no ReTiS Lab (Real-Time Systems Lab) da Scuola Superiore Sant'Anna de Pisa, Itália, tem uma arquitectura bastante versátil com capacidade para suportar tarefas hard, soft e não tempo real, usando diversas políticas de escalonamento de tarefas. São também disponibilizadas ao utilizador diversas bibliotecas de funções, entre as quais bibliotecas gráficas, que permitem desenvolver interfaces interactivos e apelativos.

Este artigo aborda o problema físico de equilíbrio de um pêndulo invertido, descrevendo seguidamente a constituição do sistema construído, terminando com uma breve análise crítica ao respectivo desempenho.

II. DESCRIÇÃO DO PROBLEMA

O problema abordado neste artigo é o de equilíbrio de um pêndulo invertido utilizando uma base de um robô móvel com dois motores independentes. Este problema requer a leitura do ângulo do pêndulo e a actuação adequada nas velocidades dos motores. O mecânismo de fixação do pêndulo possui apenas um grau de liberdade no sentido longitudinal do robô de modo que também se atacaram os dois motores em conjunto permitindo apenas movimentos lineares. O controlador utilizado para fechar a malha do sistema, isto é ler o ângulo do pêndulo e calcular a actuação necessária, recorre a uma unidade de processamento materializada num PC. Contudo, o PC tem de executar várias actividades simultâneas pois, para além do controlo do pêndulo, terá ainda de atender o teclado para permitir ajuste on-line de parâmetros e manter actualizado um interface gráfico que represente o estado instantâneo do pêndulo. Para além destas actividades, a actuação nos motores faz-se através de um sinal eléctrico com codificação PWM, com largura de impulso entre 1 e 2 ms, que tem de ser gerado de forma relativamente precisa pelo PC, num pino da respectiva porta paralela.

Para realizar esta multiplicidade de actividades com o rigor temporal necessário ao equilíbrio na vertical de um pêndulo, sistema instável, utilizou-se um kernel de tempo real, o qual permite cumprir restrições temporais associadas a cada tarefa.

O diagrama funcional do sistema está representado na Fig. 1, onde se podem identificar os dois blocos fundamentais, o sistema base móvel mais pêndulo e o sistema controlador.

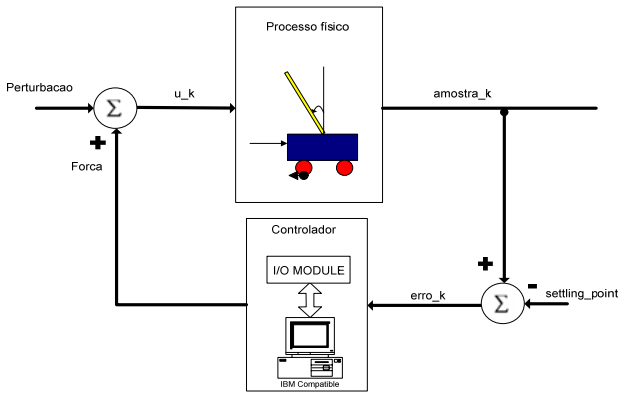

Fig. 1 – Diagrama funcional do sistema.

III. MODELO FÍSICO

Com vista a permitir construir um controlador adequado foi efectuada uma busca de modelos físicos do sistema. Com o modelo encontrado [3] foi realizado um estudo físico do sistema, nomeadamente a resposta do mesmo em malha aberta e fechada. O modelo encontrado é relativamente completo e encontra-se descrito na Fig. 2 e no sistema de equações (1), em que *M* é a massa da base, *m*, *l* e J a massa, comprimento e momento de inércia do pêndulo, e *F* a força de controlo aplicada à base.

$$\begin{cases} (M+m)\dfrac{\partial^2 x}{\partial t^2} + b\dfrac{\partial x}{\partial t} - ml\dfrac{\partial^2 \theta}{\partial t^2} = F \\ (J+ml^2)\dfrac{\partial^2 \theta}{\partial t^2} - ml\dfrac{\partial^2 x}{\partial t^2} = mgl\theta \end{cases} \qquad (1)$$

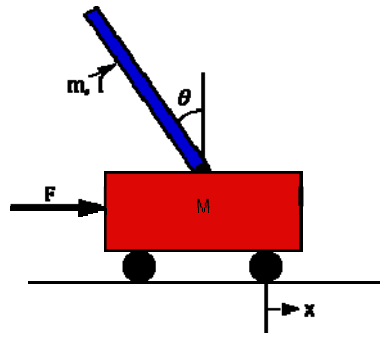

Fig. 2 – Modelo físico do sistema

IV. CONSTITUIÇÃO FÍSICA DO SISTEMA

A Fig. 3 representa os blocos físicos constituíntes do sistema. O interface ao PC é efectuado pela respectiva porta paralela, através de um circuito expressamente projectado para o efeito.

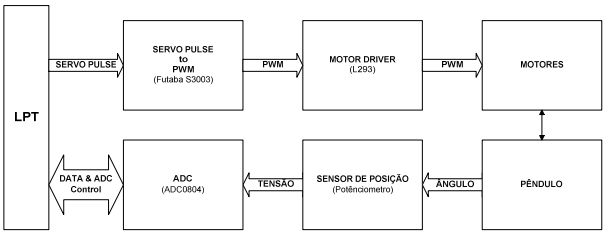

Fig. 3 – Diagrama dos blocos físicos do sistema

Este circuito de interface contém um controlador PWM retirado de um servomotor Futaba S3003 ao qual estão acopladas duas pontes H (L293) que controlam o movimento dos motores. O circuito também mede o ângulo do pêndulo através de um potenciómetro acoplado ao respectivo eixo de rotação (sensor de posição) e converte-o para um formato digital utilizando uma ADC.

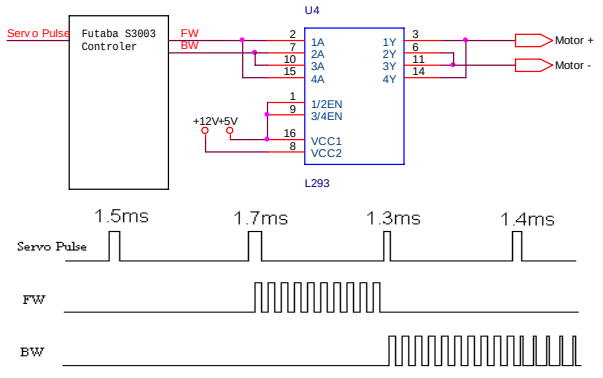

Fig. 4 – Controlo PWM, bidireccional, dos motores.

A Fig. 4 mostra o esquema (em cima) do controlo dos motores a partir da linha PWM gerada pelo PC (*servo pulse*), bem como os respectivos sinais eléctricos (em baixo) à entrada e saída do controlador S3003. O sinal *servo pulse* é gerado no PC por software com um período de 20ms e largura de impulso entre 1 e 2 ms. O centro desta variação, 1,5ms, corresponde a parar os motores, 1ms de largura corresponde à velocidade máxima num sentido e 2ms à velocidade máximo no sentido contrário.

Em relação à leitura do ângulo do pêndulo, para melhor aproveitar a resolução da ADC a respectiva gama dinâmica foi limitada a 2,5V +/-1V.

V. SOFTWARE

Em termos de estrutura do programa, utilizaram-se 5 tarefas, cada uma dedicada a uma finalidade específica. A Fig. 5 mostra um diagrama de blocos da arquitectura de software da aplicação, ilustrando as tarefas e as estruturas de dados utilizadas.

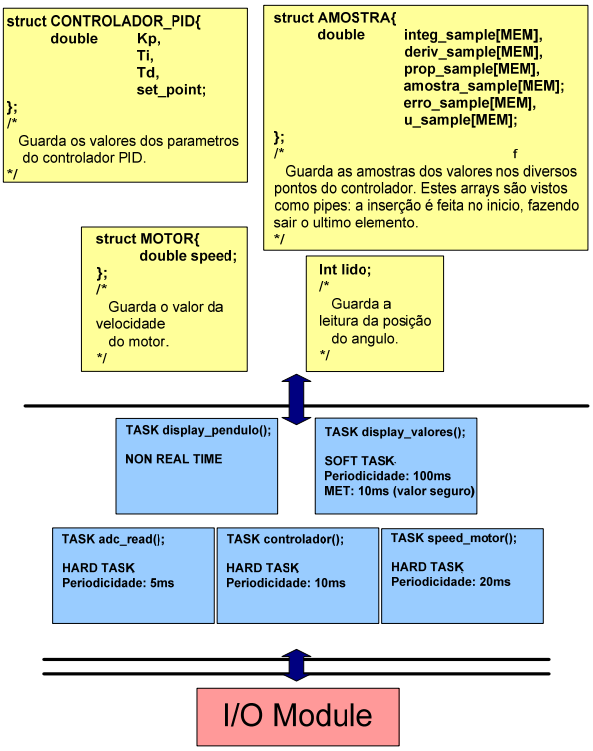

Fig. 5 – Arquitectura de software

As várias tarefas utilizadas são:

1- adc_read( )

Descrição: faz uma leitura do valor do ângulo do pêndulo à entrada da porta paralela. Esta leitura é feita recorrendo a uma biblioteca que agrupa diversas funções de leitura/ configuração/ escrita, relativas à porta paralela.

Tipo de tarefa: tarefa crítica relativamente ao sistema em causa, pelo que é considerada HARD TASK.

Periodicidade: a sua periodicidade é de 5ms. Contudo, a actuação está limitada a um período de 20ms imposto pelos controladores PWM (ver secção anterior)..

2- controlador( )

Descrição: com base nos parâmetros Kp e Td indicados pelo utilizador para o processo de controlo (coeficientes de um controlador PD), esta tarefa calcula o valor de velocidade a aplicar aos motores.

Tipo de tarefa: tarefa crítica relativamente ao sistema em causa, pelo que é considerada HARD TASK.

Periodicidade: a periodicidade escolhida para esta tarefa é de 10ms, tem que ser um valor naturalmente maior que 5ms (valor da taxa de amostragem) e menor que 20ms (limite imposto pelo hardware).

3- speed_motor( )

Descrição: esta tarefa simplesmente encarrega-se de gerar o sinal com período de 20ms, em que o degrau inicial varia entre 1ms e 2ms de forma a controlar a velocidade e direcção da rotação dos motores.

Tipo de tarefa: tarefa crítica relativamente ao sistema em causa, pelo que também é considerada HARD TASK.

Periodicidade: esta tarefa tem periodicidade de 20ms.

4- display_valores( )

Descrição: esta tarefa está encarregue do display da informação no ecrã. São mostrados os valores das constantes do controlador PD e da velocidade máxima dos motores, e ainda o ângulo instantâneo do pêndulo (Fig. 6).

Tipo de tarefa: esta tarefa não é critica para o funcionamento do sistema, sendo considerada SOFT TASK.

Periodicidade: um valor razoável para o refresh de um mostrador de valores é de 10 frames por segundo, pelo que o valor escolhido para a periodicidade desta tarefa é de 100ms.

5- display_pendulo( )

Descrição: ao cargo desta tarefa está a representação gráfica da inclinação do pêndulo (Fig. 6).

Tipo de tarefa: procurando explorar todas as potencialidades do kernel, esta tarefa foi definida como NON REAL TIME, executando em *background*.

Periodicidade: não se aplica.

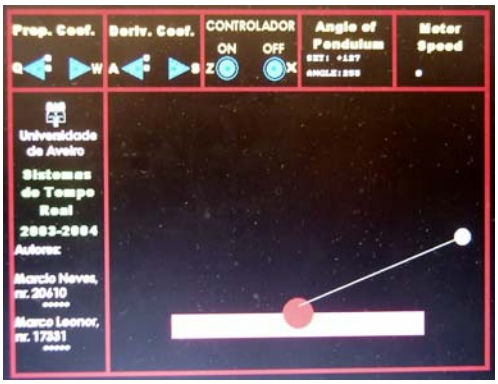

Fig. 6 – Interface gráfico do pêndulo invertido.

VI. RESULTADOS

O sistema construído funcionou da maneira prevista, conseguindo um equilíbrio bastante eficiente do pêndulo (Fig. 7). Contudo, o robô não tende a parar numa dada posição, o que é de esperar já que o sistema não tem informação sobre a posição da base móvel. O controlo utilizado foi do tipo PD, proporcional-derivativo.

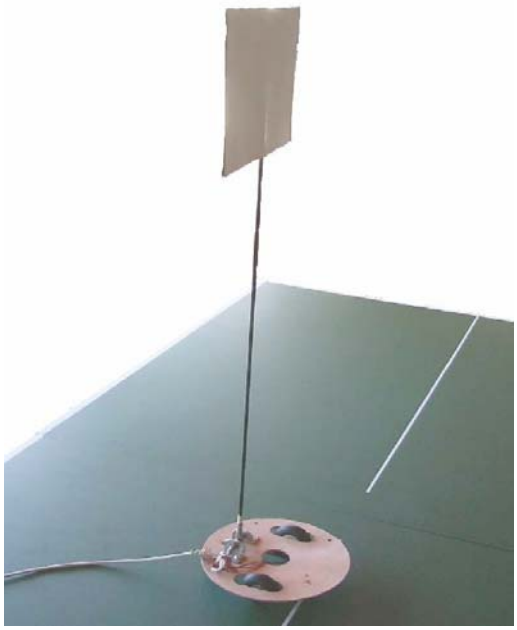

Fig. 7 – Foto do pêndulo invertido em equilíbrio.

A exitência de um interface gráfico (Fig. 6) para a visualização e controlo do processo tornou este sistema interactivo e mais educacional. Por exemplo, através da mudança dos parâmetros do controlador *on-the-fly*, foi possível verificar o respectivo impacto no comportamento do sistema. Verificou-se que o termo proporcional determina a velocidade com que o motor reage a variações do ângulo do pêndulo e o termo derivativo reduz o carácter oscilatório do sistema reduzindo assim o *settling time*.

Finalmente, verificou-se existir uma sensibilidade elevada à afinação do ponto de velocidade zero dos motores, que deve corresponder à posição vertical do pêndulo. Note-se que um pequeno desajuste para qualquer um dos lados implica uma tendência natural no carro a deslocar-se para esse mesmo lado, dificultando o controlo.

VII. CONCLUSÃO

Este artigo descreveu um sistema de equilíbrio de um pêndulo invertido sobre uma plataforma de um pequeno robô móvel. O controlo é efectuado num PC de apoio, com interface pela porta paralela. A arquitectura de software está baseada no kernel tempo real SHaRK, utilizando 5 tarefas, duas das quais permitem manter um interface gráfico interactivo. É de realçar o bom desempenho do sistema, mesmo considerando que apenas se utilizou material de muito baixo custo.

REFERÊNCIAS

[1] Luís Almeida. Página da disciplina Sistemas de Tempo-Real http://sweet.ua.pt/~lda/str/str.htm, 2003.

[2] Documentação disponível no site oficial do SHARK (http://shark.sssup.it/)

[3] Katsuhiko Ogata. Engenharia de controlo moderno. 3ª ed.. Prentice Hall do Brasil, 1998.