3. SISTEMA DE CONTROLO DE BAIXO-NÍVEL

Resumo:
Este capítulo tem como objecto de estudo os actuadores presentes nas juntas no que toca à leitura do seus sensores e à sua actuação. Posteriormente combinaremos estes dois procedimentos de forma a corrigir eventuais desvios ao comportamento esperado, pela introdução de um compensador externo. Posteriormente este controlador será utilizado nas juntas do robot responsáveis pela locomoção.

3.1. CONTROLO DA PLATAFORMA HUMANÓIDE

Como já foi mencionado no capítulo 1, o robot humanóide é constituído por oito unidades de controlo local, que denominámos por *slaves*, que podem controlar até 3 graus de liberdade dos 22 existentes na arquitectura actual, e que se interligam a um computador (unidade principal), através de uma unidade *master*, para o envio de comandos de actuação com base na informação sensorial medida.

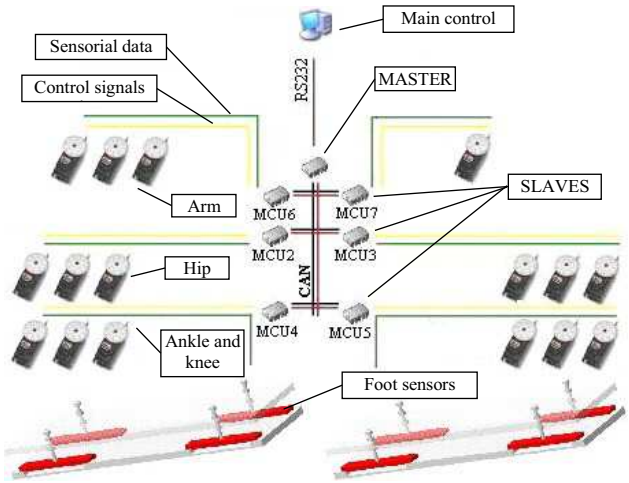

Fig. 22: Arquitectura da plataforma humanóide.

Como actuador para cada grau de liberdade, estamos a utilizar servomotores, que não são nada mais, nada menos, do que motores adaptados para efectuar controlo de posição. Como vantagens estes motores permitem-nos fazer medidas da sua posição e da sua corrente consumida o que nos possibilitará mais adiante fazer controlo externo para correcção de eventuais desvios ao comportamento ideal.

Para além do controlo sobre as juntas, também existem um conjunto de sensores adicionais destinados ao equilíbrio desta plataforma. Eles são:

- ➔ Sensores de força aplicados directamente sobre a base de cada pé, implementados a partir de extensómetros – resistências que variam o seu valor de acordo com a sua deformação.
- ➔ Inclinómetros para medição da verticalidade do tronco. Estes inclinómetros são basicamente acelerómetros que medem a aceleração da gravidade nos seus dois eixos ortogonais – na posição vertical o vector gravidade deve coincidir com o seu eixo vertical.
- ➔ Giroscópios para medição da velocidade angular de certos pontos do corpo.

Para já, ainda só foram testados os sensores de força aplicados na base dos pés, com o intuito de assegurar a projecção do centro de massa do corpo sobre o centro de um dos pés (quando apenas um pé está assente no solo). Desta forma, para pequenas velocidades e pequenas acelerações, o robot mantém-se em equilíbrio. Este assunto será discutido de forma mais detalhada no próximo capítulo.

Neste capítulo apenas serão discutidos os actuadores e o seu controlo de forma a permitir:

- ➔ Fazer o controlo de posição de modo a garantir que o actuador atinge sempre a posição solicitada;
- ➔ Fazer o controlo de velocidade, variando o seu valor de acordo com as necessidades;
- ➔ Que os movimentos das juntas sejam os mais suaves possíveis, sem acelerações bruscas nem velocidades muito elevadas;

Embora os servomotores tenham sido escolhidos por possuírem um controlador interno de posição, contrariamente aos outros géneros de motores, veremos adiante que apresenta muitos problemas na concretização dos objectivos enunciados, principalmente na presença de grandes cargas no seu eixo.

3.1.1. A Unidade de Controlo Local

A Fig. 23 apresenta a constituição de uma unidade de controlo local – *slave* – com o microcontrolador PIC18F258 e a electrónica de interface para os servomotores e os diversos sensores adicionais.

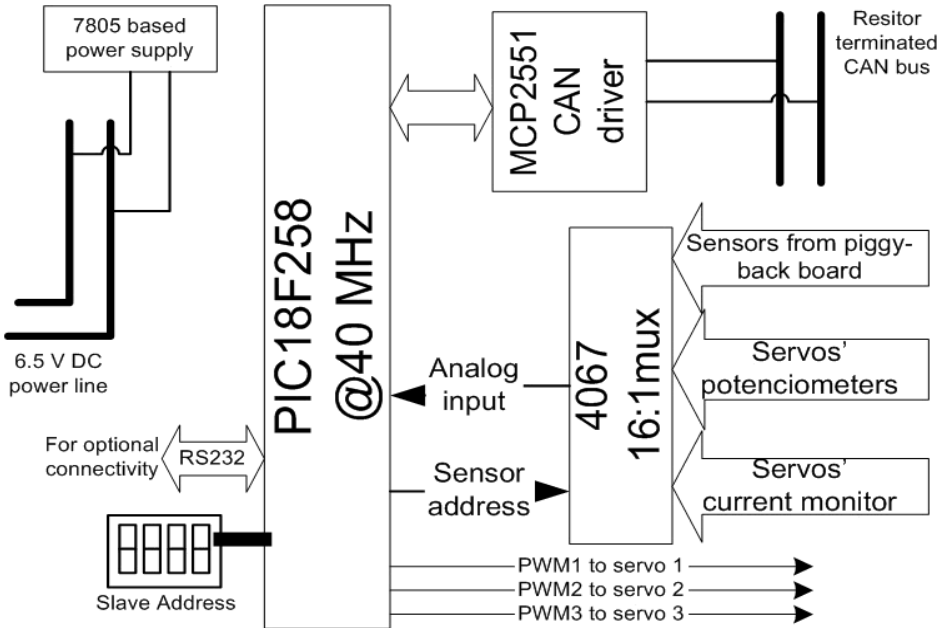

Fig. 23: Constituição de uma unidade slave.

Como se pode observar, cada slave, possui uma interface para o barramento CAN constituído pelo MCP2551 *CAN driver* uma interface adicional para comunicar pela USART e um *DIP switch* que permite via hardware configurar o endereço da unidade (*slave address*). Para a actuação, é possível controlar três servomotores com a disponibilidade de um pino dedicado a cada um para aplicação directa de um sinal modulado em *duty-cycle* para o controlo de posição. Adicionalmente, um multiplexer de 16 canais é utilizado para fazer a aquisição sensorial dos servomotores e dos sensores especiais.

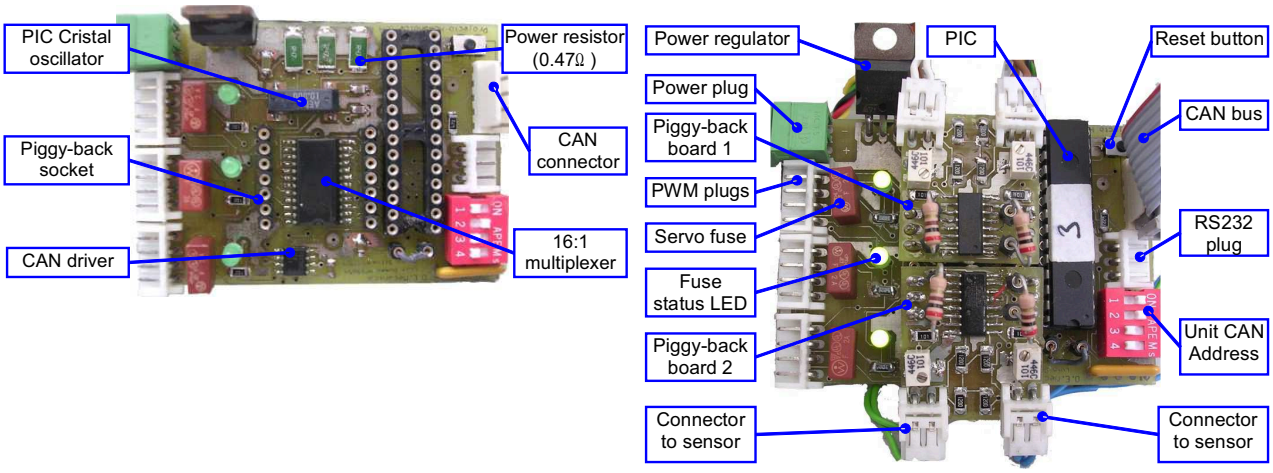

Fig. 24: Imagens de uma unidade de controlo local.

As imagens da Fig. 24 correspondem à versão implementada no ano lectivo anterior (2004/05). No que respeita a este ano, modificações foram feitas de modo a resolver alguns bugs. Elas são:

- ✔ Fusíveis de 2A substituídos por de 3.15A;

  Devido aos picos de corrente induzidos pelo arranque dos servomotores, provou-se ser insuficiente a corrente máxima de 2A, pelo que foi necessário aumentar para um escalão acima: 3.15A.

- ✔ Condensador electrolítico de 1µF adicionado à entrada do regulador de tensão;

  Para evitar a ocorrência de resets nos PICs adicionou-se um condensador de filtragem à entrada do regulador de tensão, para estabilização da tensão de entrada.

- ✔ Condensador SMD de 100nF colocado aos terminais da alimentação do PIC;

  De modo a conferir maior estabilidade na alimentação do PIC colocou-se um condensador de filtragem directamente sobre os pinos da alimentação. Por uma questão de boas práticas é recomendado colocar um condensador de filtragem de elevado valor à entrada do regulador e um condensador de menor valor directamente sobre os pinos de alimentação de cada circuito integrado que esteja ligado à saída do regulador. Por isso, caso uma nova versão seja construída no futuro, aconselha-se a implementação deste método.

- ✔ Remoção das resistências de potência ligadas entre a massa dos servomotores e a do SCU;

  Com a presença desta resistência, verificava-se que os níveis de corrente consumidos eram bastante maiores que na sua ausência. Como estas resistências eram utilizadas para medição de corrente, e tendo em conta que se descobriu outro método para efectuar este procedimento, estas resistências foram curto-circuitadas (ver secção 2.1.3.1).

- ✔ Adicionadas resistências de 10KΩ entre a entrada de actuação de servomotor e a massa.

  Por motivos desconhecidos, alguns servomotores, na ausência do sinal de entrada de actuação, ficavam descontrolados efectuando movimentos contínuos sem uma posição final específica. Descobriu-se que ligando uma resistência à entrada do sinal de actuação com o outro terminal à massa, este efeito deixava de se verificar.

3.2. ACTUADORES: OS SERVOMOTORES

Para a actuação sobre as juntas é essencial tanto o controlo de posição como de velocidade, e dado que em média cada junta não possui uma excursão de movimento superior a 180º, uma solução baseada em servomotores foi a mais imediata. Podem-se enunciar as seguintes vantagens e desvantagens para esta escolha:

- ✔ Excursão de posição de 180º;
- ✔ Controlador de posição incluído;
- ✔ Relativamente pequeno e compacto;
- ✔ Relativamente barato;
- ✗ Não oferece controlo de velocidade.

Fig. 26: Servomotor da HITEC

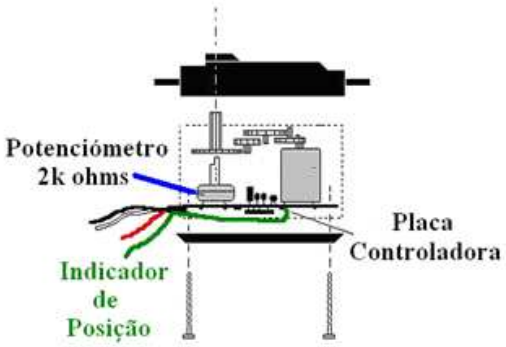

Fig. 25: Representação do interior de um servomotor.

Olhando para o interior deste actuador podemos discernir os seguintes componentes:

- ➔ Motor DC;
- ➔ Caixa redutora;
- ➔ Mecanismo de *feedback* da posição angular do motor (potenciómetro);
- ➔ Electrónica de controlo do motor a partir de um sinal digital;
- ➔ Electrónica de controlo de posição.

Pode-se, por isso, concluir que estes servos, não são nada mais do que simples motores DC que incluem electrónica interna responsável por implementar o controlo de posição em malha fechada a partir de um sinal externo de referência. Este sinal externo define a posição a ser atingida é da forma de uma onda quadrada modulada em largura de impulso – PWM (*Pulse Width Modulation*) –, cuja largura é que define a posição final. Desta forma, a posição do eixo do servo é controlada a partir do *duty-cycle* de uma onda quadrada de formato digital (Fig. 27).

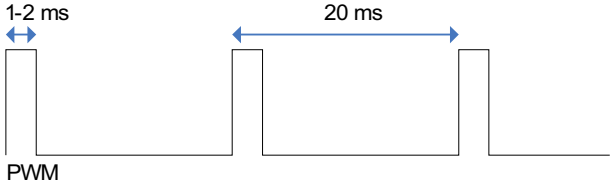

Fig. 27: Sinal de PWM aplicado no servomotor.

Spec	Values	
Control system	Pulse Width Control 1.5 ms neutral	
Voltage range	4.8V to 6.0V	
Teat voltage	@ 4.8V	@ 6.0V
Speed (no load)	60º/0.19 s	60º/0.14 s
Stall torque	1.94 Nm	2.42 Nm
Operating angle	45º /one side pulse traveling 400µs	
Direction	clockwise/pulse traveling 1.5 to 1.9 ms	
Current drain	8mA (idle); 700mA (no load running)	
Dead bandwidth	8 µs	
Dimensions	66 x 30 x 57.6 mm	
Weight	152g	

Tabela 28: Especificações do servo da HITEC HS-805BB.

Para cada modelo de servomotor este sinal deve assumir determinadas especificações, mas para um dos modelos utilizados (HITEC HS-805BB[1]) estas devem ser as características do sinal de PWM:

- ➔ Frequência de 50 Hz (Período de 20 ms);
- ➔ Duty-cycle variável entre 1 (0º) e 2 ms (180º).

Mais dados podem ser consultados na Tabela 28 com indicações do *duty-cycle* correspondente à posição neutra ou central (90º) (*control system*), gama das tensões do sinal de PWM (*voltage range*), em que observa que aceita sinais digitais de 5V provenientes do PIC, binário máximo, corrente consumida, e outros.

A escolha do modelo HITEC HS-805BB para as juntas mais exigentes, recaiu essencialmente pelo seu binário máximo de 2.42 N.m. Simulações em CATIA da plataforma humanóide realizando um passo (ano 2004/05) demonstram que no pior cenário as juntas podem estar submetidas a binários de cerca de 2.6 N.m (Tabela 29). O modelo HS-805BB destacou-se por possuir o máximo binário da gama disponível, que mesmo assim, mostra ser insuficiente para o nosso caso. Para resolver este problema, introduziram-se correias de transmissão que multiplicam o binário máximo pela relação entre número de dentes do eixo do servo e da junta.

Motor / Junta	Θ_1 [º]	T_1 [N.m]	Θ_2 [º]	T_2 [N.m]	Θ_3 [º]	T_3 [N.m]
Pé 1 lateral	0.0	**2.37**	7.1	0.98	7.1	0.96
Pé 1 frente	4.7	0.30	10.1	0.20	10.1	0.04
Joelho 1	10.1	0.76	21.8	1.17	21.8	1.01
Anca 1 frente	5.4	0.35	11.7	0.30	11.7	0.14
Anca 1 lateral	0.0	**2.26**	7.1	**2.57**	7.1	**2.55**
Pé 2 lateral	0.0	0.00	7.1	0.00	7.1	0.00
Pé 2 frente	4.7	0.12	10.1	0.12	16.4	0.12
Joelho 2	10.1	0.17	21.8	0.23	41.9	0.30
Anca 2 frente	5.4	0.07	11.7	0.02	25.6	0.14
Anca 2 lateral	0.0	0.01	7.1	0.30	7.1	0.29

Tabela 29: Binários exigidos na simulação de um passo.

$$N = \frac{Número\ de\ dentes\ da\ polia\ da\ junta}{Número\ de\ dentes\ do\ eixo\ do\ servo}$$

$$\tau_{máx} = N \times \tau_{servo}$$

Fig. 28: Correia de transmissão aplicada a um servo.

Nas juntas dos pés e dos joelhos utilizaram-se relações de transmissão de 2:1 e de 2.2:1 assegurando um binário máximo de 5.3 N.m assumindo como 2.42 o binário máximo do servomotor, e nas juntas das ancas aumentou-se a margem para 1:3.75 dada a sua exigente natureza (máximo de 9.1 N.m).

1 Estes dispositivos podem ser adquiridos no site http://www.maxxprod.com

Olhando para o interior do servo na tentativa de perceber melhor a sua electrónica interna, a Fig. 29 apresenta o circuito de controlo de um servo semelhante aos da HITEC – o FUTABA S3003 – cujo sinal de PWM (input pulse) apresenta as mesmas especificações que as descritas atrás.

Estes dispositivos utilizam como *feedback* tanto o sinal de posição como de velocidade que, além de fornecer um controlo preciso de posição, permite estabilizá-lo de modo a conduzi-lo para a posição desejada com o mínimo de oscilação.

Em funcionamento normal, o sinal de PWM da entrada é comparado com o sinal resultante de um gerador de impulso linear controlado a partir da posição obtida pelo potenciómetro e da velocidade medida a partir da força contra-electromotriz do motor (tensão gerada entre impulsos de potência). O sinal gerado é da mesma forma que o de entrada e, para baixas velocidades, a sua largura de impulso deve corresponder à posição efectiva do motor.
A diferença de largura de impulso entre estes dois sinais, conhecido como sinal de erro, é em seguida amplificado através de um amplificador de impulso que depois é aplicado numa ponte H (BAL6686) que controla o motor. Facilmente se percebe que quando a largura de impulso do sinal de entrada é igual à resultante pelo gerador de impulso (erro zero) a diferença é de largura nula e nenhum sinal é aplicado ao motor deixando-o em repouso – note que o motor em si é controlado em velocidade pelo que se nenhum sinal for aplicado ele tende para o repouso.

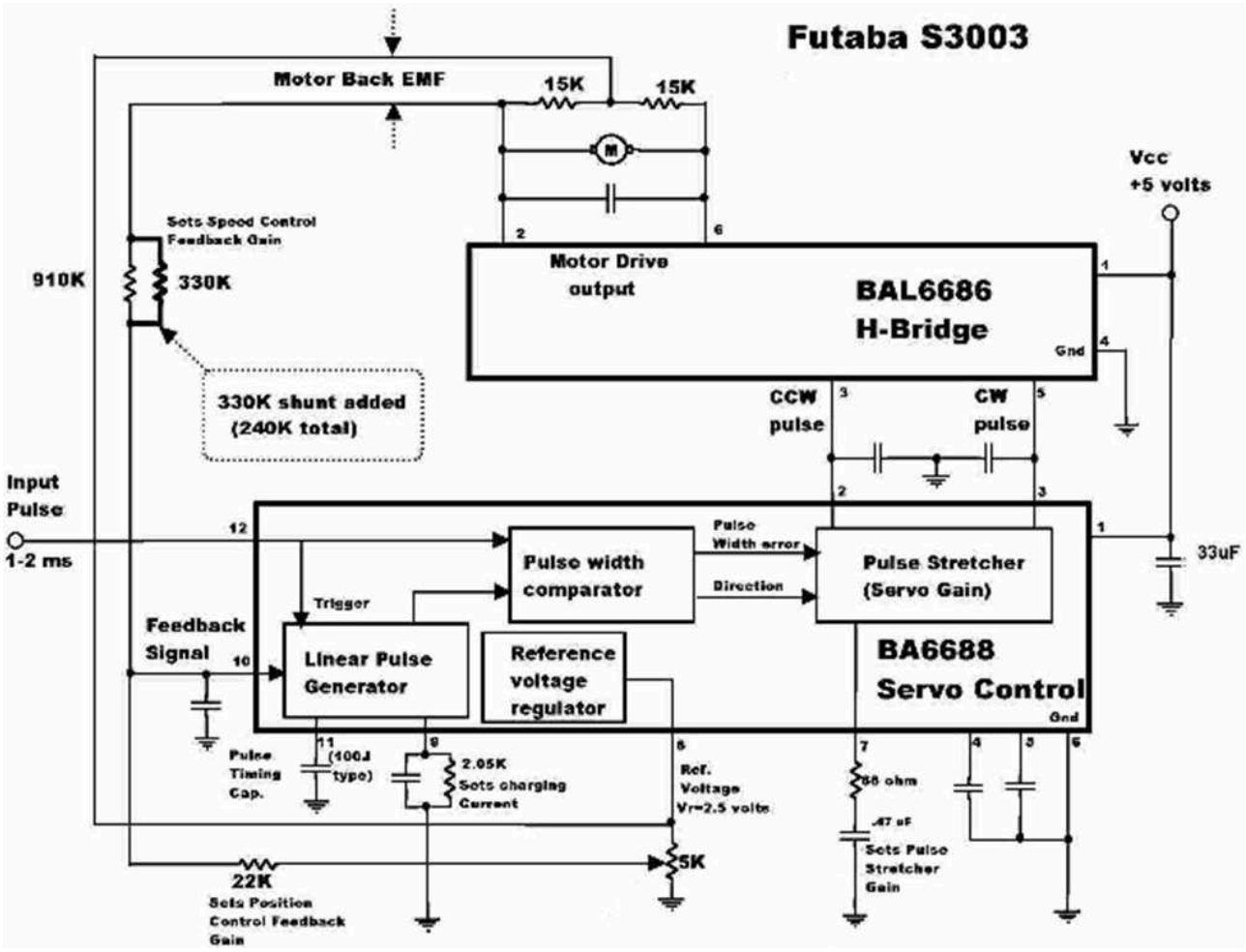

Fig. 29: Circuito do controlador de posição de um FUTABA S3003.

Representando matematicamente as operações envolvidas, verificamos que:

➔ À posição medida pelo potenciómetro p(t) é aplicado um ganho K_P representado pela resistência de 22KΩ que é somada à velocidade resultante da força contra-electromotriz multiplicado pelo ganho K_D representado pelo paralelo das resistências de 910K e de 330KΩ.

$$F(s)=K_P\cdot P(s)+K_D\cdot V(s)$$

$$F(s)=(K_P+s\cdot K_D)\cdot P(s)$$

➔ O resultado da soma f(t) é introduzido no gerador de impulso linear que produz um sinal de PWM para comparação (em termos de largura de impulso) com o sinal de entrada de referência r(t) através do comparador de largura de impulso produzindo o sinal de erro e(t) também do formato PWM.

$$E(s)=R(s)-F(s)$$

➔ O sinal de erro é amplificado num factor de K através de um amplificador de impulso e é aplicado ao motor através de uma ponte H.

$$U(s)=K\cdot E(s)$$

Logo, concluindo:

$$\boxed{U(s)=K\cdot[R(s)-P(s)\cdot(K_P+s\cdot K_D)]}$$

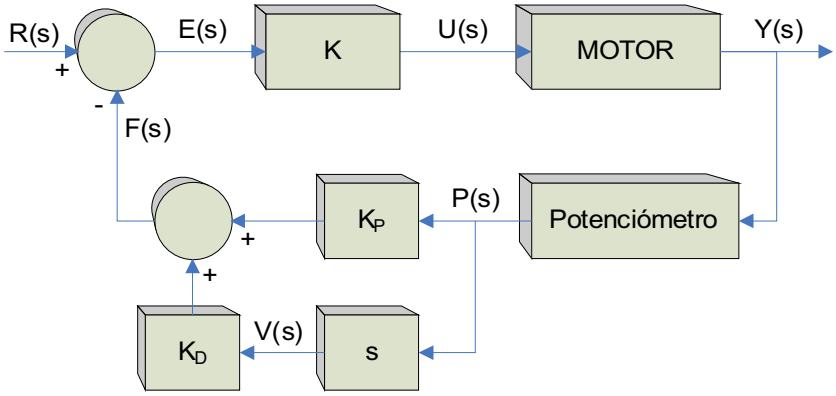

Fig. 30: Representação esquemática do controlador de posição interno.

Comparando a Fig. 29 e a Fig. 30 podemos relacionar o elemento *s* como o sensor de velocidade, a diferença R(s) – F(s) como o comparador de largura de impulso e o ganho K como o amplificador de largura de impulso. Verifica-se então dois tipos de compensação presentes: a compensação série executada pelo ganho proporcional K e a compensação paralela realizada na realimentação através de um ganho PD (proporcional+derivativo) à posição medida. Note que o sinal proveniente da realimentação f(t) para posterior comparação não depende exclusivamente da posição medida, mas também da velocidade actual o que confere uma maior estabilidade na realização do percurso para a posição desejada.

Um pormenor que vale a pena salientar é a ausência da componente integral no controlo, o que poderá originar erros em regime estacionário quando aplicadas cargas elevadas no eixo.

3.2.1. Setup Experimental

Para efeitos de testes ao actuador em questão, uma base experimental foi montada tendo em vista a realização de movimentos variando os seguintes parâmetros:

- ➔ Excursão do movimento (posição inicial e final);
- ➔ Velocidade do movimento[2];
- ➔ Carga aplicada no eixo.

... de forma a poder avaliar a performance do servomotor pela monitorização da posição efectiva que percorre ao longo do tempo e da corrente consumida.

Fig. 32: Setup experimental.

Load	Mass (g)	Torque (N.m)
0	9	0.009
1	258	0.253
2	463	0.454
4	675	0.662
1+4	924	0.906
2+4	1129	1.108
1+2+4	1378	1.352

Tabela 30: Lista de cargas utilizadas para teste.

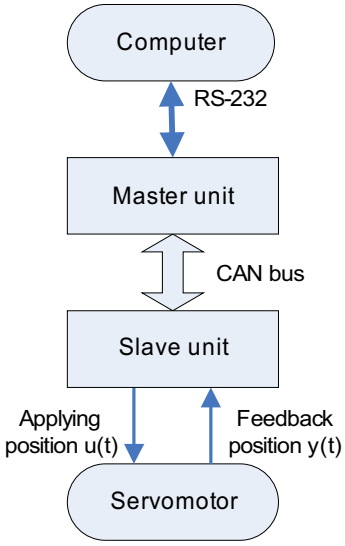

Fig. 31: Arquitectura das comunicação no setup.

Pela Fig. 31 podemos visualizar a arquitectura das ligações com a unidade principal (PC). Como só tencionámos testar um único actuador, apenas precisamos de uma unidade *slave* que liga ao PC segundo a rede de comunicações utilizada na plataforma humanóide.

O servomotor, por sua vez, encontra-se fixo num torno de fixação para poder mover cargas de uma forma segura. As cargas são presas ao servo através de um braço de massa desprezável e de 10 cm de comprimento que percorre a excursão dos 180º desde a posição vertical inferior até ao ponto vertical superior. Esta opção é muito útil dado que nos permite estimar o valor do binário resultante da força gravítica através da seguinte fórmula...

$$\tau = F_G * L * \cos(\theta) = m * g * L * \cos(\theta)$$

... em que m é a massa da carga, g a aceleração gravítica (9.81 m/s^2), L é o comprimento do braço (0.1m) e θ é a posição angular do braço, com 0º na posição central (perpendicular ao vector gravidade), +90º no extremo superior e -90º no extremo inferior. A partir dela sabemos que nos extremos o binário gravítico é nulo e no ponto central (braço e F_G perpendiculares) ele é máximo.

A Tabela 30 indica a lista de massas utilizadas durante os testes. Embora as especificações destes servos indiquem um binário máximo de 2.42 N.m, na prática verificou-se que este estava muito abaixo deste valor deixando de responder para cargas superiores a 1.5 Kg (binário τ=1.47 N.m) talvez devendo-se a desgaste destes actuadores. Tal justifica o facto da massa mais elevada da lista ser de 1.38 Kg.

2 Embora não seja possível fazer controlo directo de velocidade, é possível a partir do controlo de posição induzir uma determinada velocidade ao dispositivo. Tal é explicado no capítulo seguinte.

3.2.2. Actuação sobre o Servomotor

Tal como foi descrito na secção 3.1 cada unidade de controlo local é capaz de gerar três sinais de PWM independentes entre si, a partir do microcontrolador PIC disponibilizando-os através de três pinos reservados que se podem ligar directamente aos servomotores. Note, contudo, a necessidade de adicionar a cada saída de PWM uma resistência de 10KΩ à massa para drenagem da carga existente aquando da ausência de PWM. Sem este elemento, nestas circunstâncias, o servo poderá deslocar-se à máxima velocidade para posições imprevistas que podem não se encontrar na gama anunciada dos 180º resultando no encrave num dos extremos com o máximo consumo de corrente. Tal comportamento imprevisto pode danificar estes dispositivos não mencionando a própria estrutura do humanóide.

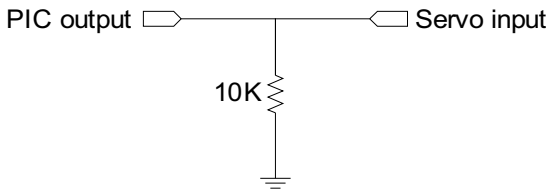

Fig. 33: Resistência a adicionar a cada saída de PWM do PIC.

Vamos agora descrever como é programado o PIC para gerar os três sinais de PWM. Tendo em mente a geração destes sinais de forma automática sem nunca colocar o CPU em espera (por *polling*) várias opções são-nos oferecidas, nomeadamente utilizando...

- ➔ Módulo de PWM (CCP);
- ➔ Módulo de comparação (CCP);
- ➔ Interrupções.

Dado o PIC 18F258 só possuir dois módulos CCP (CCP e ECCP) tal é insuficiente para gerar três sinais de PWM, com a agravante da frequência mínima configurável para o módulo de PWM ser bastante superior a 50 Hz. Por isso só nos resta a última opção recorrendo unicamente a interrupções baseadas em *timers*.
Só pensando na geração de um único PWM bastaria o uso de dois *timers*: um para a elevação do sinal a 1 com uma frequência fixa de 50 Hz e um segundo para a descida do sinal a zero depois de um determinado período de tempo após a elevação. Desta forma obtém-se um sinal de PWM cujo período de tempo a 1 corresponde ao *duty-cycle* desejado para definição da posição do servo.

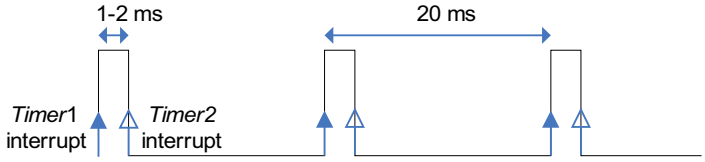

Fig. 34: Geração de um sinal de PWM através de dois timers.

A Fig. 34 esquematiza este procedimento fazendo uso dos *timers* 1 e 2:

1. O *timer* 1 é programado para gerar uma interrupção a cada 20ms (50Hz de frequência):
   A cada instanciação o pino de PWM é elevado a um;
2. Após cada interrupção do *timer* 1, o *timer* 2 é programado para gerar uma segunda interrupção após um período de tempo correspondente ao *duty-cycle* desejado (entre 1 e 2 ms):
   Após a ocorrência desta interrupção, o pino de PWM é baixado a zero.

No entanto, importa escalar este procedimento para a geração de três sinais de PWM que, embora todos tenham a mesma frequência, o *duty-cycle* gerado para os três servos devem ser independentes entre si. Uma solução poderia ser o uso de *timers* extra, mas dada a limitação de recursos, desenvolveu-se um método que permite o controlo de *N* servos usando apenas estes dois *timers*, cuja quantidade *N* apenas depende da velocidade de processamento do CPU:

1. O *timer* 1 é programado para gerar uma interrupção a cada 20ms (50Hz de frequência):

A cada instanciação o pino de PWM é elevado a um;

2. Após cada interrupção do *timer* 1, o *timer* 2 é programado para gerar uma segunda interrupção após um período de tempo correspondente ao *duty-cycle* mínimo: 1ms.
   Na ocorrência da primeira interrupção, o timer 2 é reprogramado para gerar interrupções de alta frequência de periodicidade correspondente à variação mínima de posição do servo Δ_{pos}. Para um passo de 1º, o período será:

$$T_{hf} = \frac{duty_{max} - duty_{min}}{180º/\Delta_{pos}} = \frac{2000\,\mu s - 1000\,\mu s}{180º/1º} = \frac{1000\,\mu s}{180} = 5.56\,\mu s$$

3. A cada instanciação da interrupção de alta frequência (passo de 5.56µs), a cada servo é verificado se o PWM correspondente deve baixar a zero nessa iteração. Em caso negativo, nada é feito e a próxima iteração é aguardada para uma nova avaliação.
   Na última iteração (correspondente ao *duty-cycle* máximo) todos os sinais de PWM devem estar a zero e o *timer* 2 é desligado. O processo é repetido na próxima interrupção do *timer* 1.

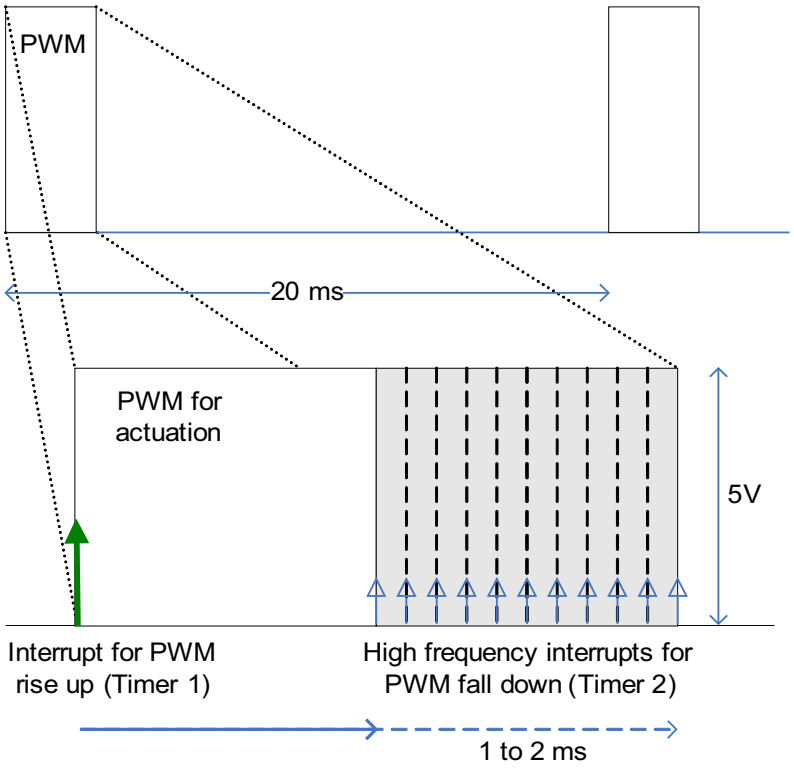

Fig. 35: Organização temporal das interrupções na geração do PWM.

A Fig. 35 demonstra bastante bem este processo. A cada servomotor é atribuído uma variável indicadora da largura de impulso a aplicar, que nada mais é o número de iterações durante as interrupções de alta frequência a manter o sinal a 1. Após a primeira interrupção um contador é utilizado para contagem das iterações que vão decorrendo entre o período de 1 a 2 ms e em cada uma delas é comparado com a variável atribuída a cada servomotor da duração do impulso. Quando o contador for igual a essa variável, o sinal de PWM correspondente é baixado a zero. Desta forma a *N* servomotores são utilizadas *N* variáveis, sendo que a única limitação é o CPU ter tempo de verificar todos os sinais de PWM e aplicar quaisquer modificações dentro do intervalo T_{hf}=5.56µs, pelo que a quantidade máxima de servos é dependente da velocidade do CPU. Para uma frequência de relógio de CPU de 10MHz a quantidade máxima de instruções *assembly* que podem ser executadas em cada iteração, é por isso:

$$N_{max} = \frac{T_{hf}}{1/f_{CPU}} = T_{hf} \cdot f_{CPU} = 5.56\,\mu s \cdot 10\text{MHz} = 55 \;\; instruções$$

Esta quantidade máxima não nos dá grande liberdade para o cumprimento da *deadline* dos 5.56µs, pelo que ou aumentamos a velocidade de CPU, ou diminuímos o número de servos a controlar, ou então, na

impossibilidade de modificar estes parâmetros, aumentamos o período T_{hf} pelo aumento do passo de posição Δ_{pos}.

Para o nosso caso em concreto, pretendemos controlar três motores utilizando um CPU com 10MHz de velocidade, pelo que só possuímos, para avaliar cada servo, cerca de 18 instruções. Como para além do processamento também entram nas contas o atraso de atendimento à interrupção, mais o código extra executado até chegar à secção de código de interesse, este limite é facilmente violado. Dada a impossibilidade de aumentar ainda mais a velocidade de CPU nem de querer aumentar o período T_{hf} sob pena de perda de resolução do servo, reestruturou-se a forma de atendimento às interrupções de forma a eliminar o tempo extra resultante do tempo de atendimento à interrupção mais o código extra. Tal foi conseguido fazendo com que o atendimento às interrupções durante o período de descida do PWM, fosse feito por *polling* dentro da própria RSI, em vez do procedimento normal de saída e reentrada nesta rotina.

A Fig. 36 apresenta o algoritmo adoptado em que, na ocorrência da primeira interrupção do *timer* 2 a entrada na RSI é feita, e só volta a sair dela após o atendimento por *polling* de todas as interrupções de alta frequência até ao fim da zona de descida do PWM. Desta forma garante-se que em cada iteração, o CPU só se dedica ao bloco de código de avaliação da descida de PWM conseguindo cumprir assim a deadline imposta dos 5.56µs.

A única desvantagem desta técnica prende-se com a impossibilidade de execução simultânea de outras tarefas durante este período. No entanto tal não é problemático dado que só corresponde a 5% de tempo de cada período de PWM.

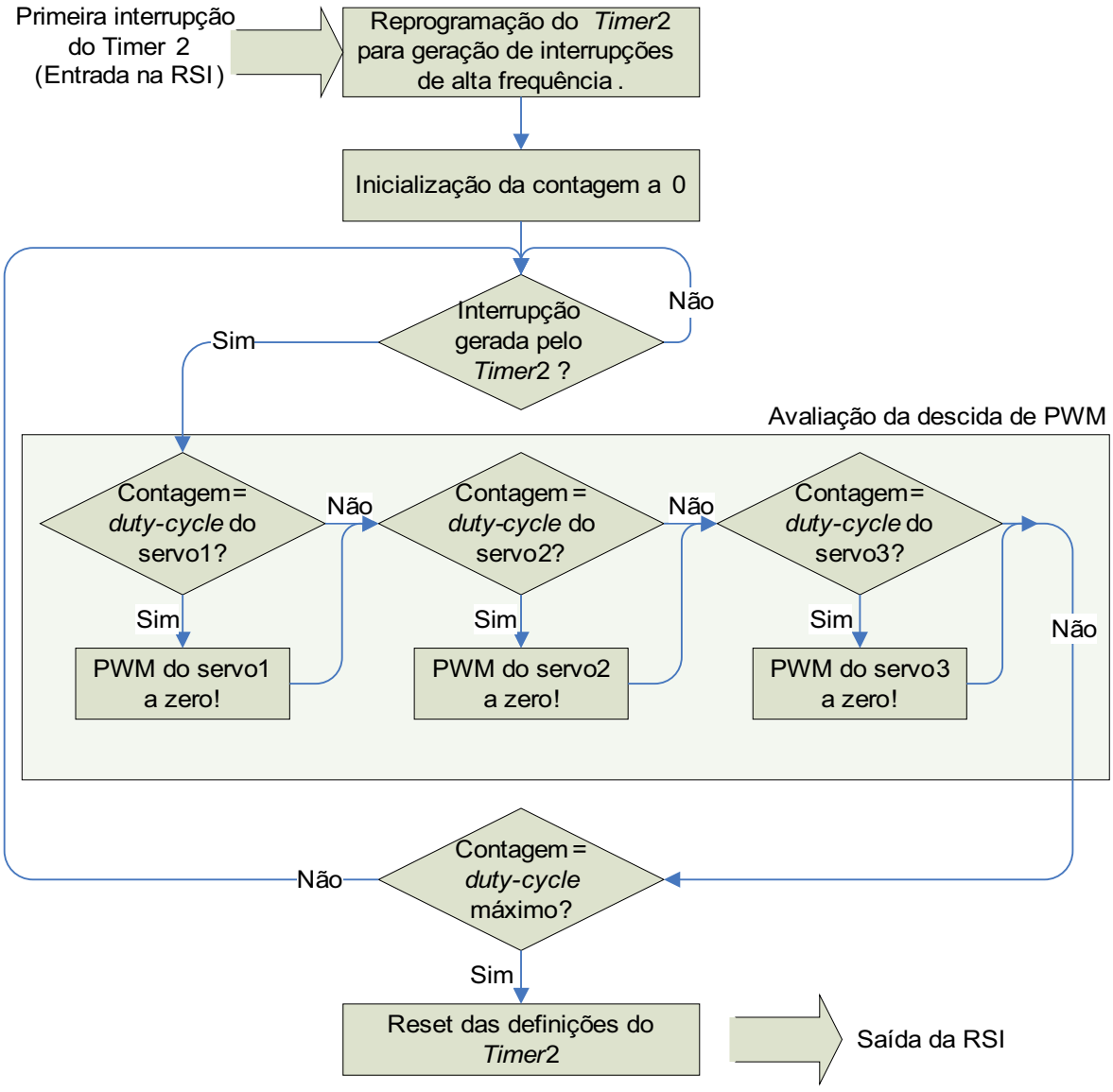

Fig. 36: Atendimento às interrupções de alta frequência.

3.2.3. Leitura Sensorial do Servomotor

Embora, por defeito, o modelo escolhido não disponibilize externamente os sinais de posição e de velocidade para *feedback*, é possível extrair o sinal de posição dado o fácil acesso ao potenciómetro interno. Desta forma, abriram-se todos os servomotores e adicionou-se um fio extra ligado ao terminal do potenciómetro indicador da posição angular do eixo (Fig. 25). Já quanto à velocidade desconhece-se a forma de aceder a este dado. Tendo acesso à informação da posição, é possível monitorar constantemente a posição do servo a partir da unidade de controlo local e transmiti-la ao PC através da rede de comunicações, podendo avaliar continuamente a performance de cada actuador.

Deslocando o servo, na ausência de carga, verifica-se uma variação da tensão de saída entre 0.8 e 1.8 V ao longo dos 180º de excursão, podendo ser amostrado pelo PIC através da ADC. Utilizando as tensões de referência *standard* para a ADC (0 e +5V) precisamos de pelo menos 10 bits para o quantificador, tendo em conta que necessitamos de uma resolução que permita distinguir 180 posições possíveis (resolução de 1º).

$$bits_{quant} = ceil\left[\log_2(níveis\ de\ quantificação)\right] \qquad níveis\ de\ quantificação = \frac{5\text{V}}{(1.8\text{V}-0.8\text{V})/180^\text{º}} = 900$$

$$bits_{quant} = ceil\left[\log_2(900)\right] = 10\ bits$$

Como a ADC do PIC oferece-nos a opção de quantificação a 10 bits tal é conveniente aproveitar.

No entanto, a medição não é tão simples como amostrar a tensão de saída quando desejado, pois na presença de cargas no eixo e/ou de velocidades elevadas surge um estranho impulso acima do nível de tensão correspondente à posição, pelo que se a medição for executada no momento do impulso o resultado será falso. Tal motivo é devido ao facto de nos modelos da HITEC, contrariamente aos da FUTABA como se verifica na Fig. 29, a tensão de referência (massa) do potenciómetro não é a mesma que para a ADC do PIC, pelo que, embora para a electrónica de controlo interna este sinal corresponderá sempre à posição do servo, para a perspectiva da ADC, este sinal possuirá oscilações na forma de impulsos ao longo do tempo, mesmo sem variar a posição do servo.

Testando diversas situações através do osciloscópio concluímos que estes impulsos possuíam propriedades não casuais:

- ➔ O impulso ocorre acima da tensão indicadora da posição;
- ➔ A amplitude do impulso é constante e apenas varia com a tensão de alimentação;
- ➔ Período de repetição coincidente com o PWM aplicado (50Hz);
- ➔ Ponto de início sincronizado com o fim do impulso de PWM;
- ➔ Largura dependente da carga e da velocidade.

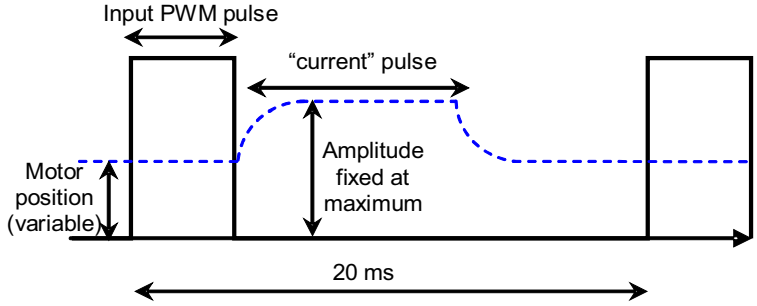

Fig. 37: Impulso de tensão medido na presença de cargas/velocidades elevadas.

O último aspecto foi o que se revelou mais interessante: quando se aumentava a carga aumentava-se a velocidade, a largura deste impulso aumentava. Tentando relacionar a largura deste impulso com a massa da carga efectuou-se o seguinte teste: fazia-se deslocar o servo para um conjunto de posições entre -90º e +90º, e para cada uma delas o actuador era deixado em repouso e media-se a largura do impulso. Desta forma, assegurávamos que a largura medida apenas era devida à gravidade (sem a interferência da velocidade). Esta experiência foi executada para duas cargas de massas diferentes, uma aproximadamente dupla da da outra.

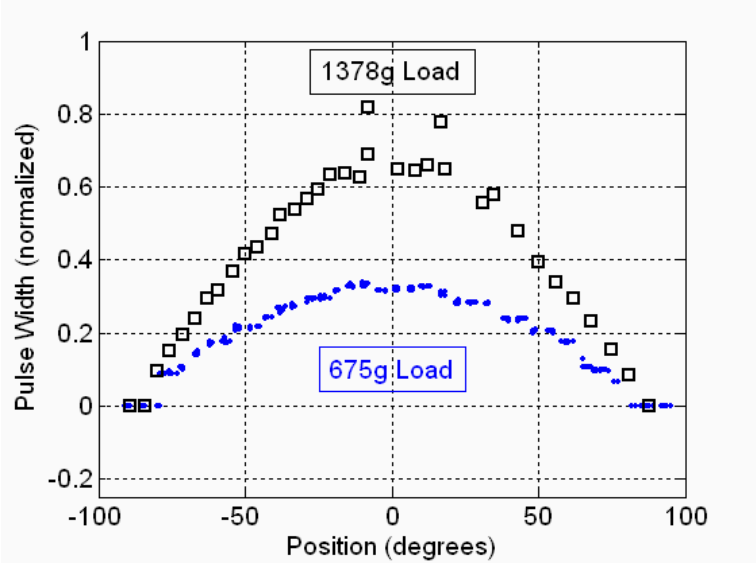

Fig. 38: Relação entre a posição e a largura de impulso.

Analisando cada carga à parte, verifica-se o comportamento sinusoidal ao longo das diversas posições sugerindo a relação com a função *seno*. Ora pela Equação 1 se conclui que a largura do impulso de tensão está directamente relacionada com o binário gravítico. Relacionando os valores entre as duas cargas, tal fundamento é reforçado dada a relação de multiplicação entre os dois conjuntos – aproximadamente de dois! Ora como se sabe a corrente está directamente relacionada com o binário...

$$I = K * \tau$$

... pelo que a largura de impulso é um indicador da corrente drenada pelo servo. Por esta razão, de agora em diante, este impulso será denominado por impulso de corrente. Este comportamento confirma a actuação sobre o motor DC de uma forma digital, também utilizando sinais no formato de PWM através da ponte H.

Note que, pelas características do sinal de posição, é possível medir tanto a posição como a corrente a partir desta única fonte, sem a necessidade de qualquer electrónica adicional para a medição de corrente!

3.2.3.1. Medição de Corrente

No ano anterior sugeriu-se um método para a medição de corrente baseado no uso de uma resistência de baixo valor em série com a alimentação do servo.

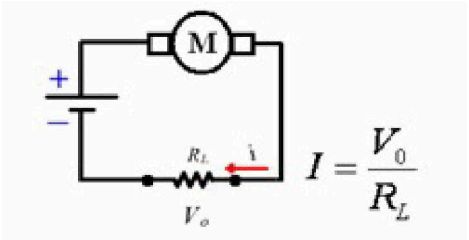

Fig. 39: Configuração possível para medição da corrente.

Embora esta forma permita uma leitura de corrente mais simplificada e de resultado sempre disponível (sem a necessidade de algoritmos de medição de largura de impulsos) acrescenta muitos mais inconvenientes do que vantagens:

- ✗ Emissão excessiva de calor nas situações de maior consumo de corrente.
- ✗ A presença da resistência provoca um aumento da corrente consumida pelo servo. Tal é explicado pelo facto de na resistência se verificar uma queda de tensão o que resulta numa diminuição da tensão de alimentação do servo, que será tanto maior, quanto maior for a corrente consumida. Com

esta diminuição, o servo tem de realizar um maior esforço o que equivale a aumentar a corrente drenada.

- ✗ Dada a queda de tensão na resistência, a tensão de referência (massa) do servo varia de acordo com a corrente consumida, o que interferirá na medição de posição uma vez que a ADC não usa a mesma tensão de referência. Desta forma, além do sinal de posição do potenciómetro ser afectado de impulsos adicionais, também será afectado pelo *offset* introduzido pela queda de tensão da resistência.

O último ponto sobressai-se pelo facto de "estragar" o sinal de posição e exigir também a medição da tensão referência do servo para compensar a variação: simplifica na medição de corrente, mas dificulta na de posição.
Em alternativa, optou-se por remover estas resistências pelo seu curto-circuito nas placas *slave* e recorrendo a ferramentas baseadas em *software* medir a corrente pela determinação da largura do impulso de corrente.

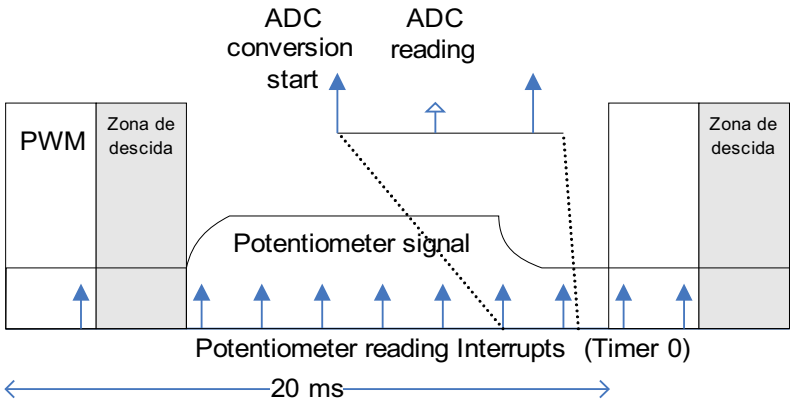

Fig. 40: Organização das interrupções (setas) na medição sensorial.

A Fig. 40 descreve bem a estratégia adoptada. Uma interrupção baseada no *timer* 0 é gerada durante todo o período de PWM com uma periodicidade de 200µs. Apenas na zona de descida do PWM ela é inibida não se pretendendo interferir com as interrupções de alta frequência. Sempre que o *timer* 0 gera uma interrupção, a tensão de saída do potenciómetro é medido e é comparado com um determinado limiar logo após o "*ADC reading*". Se estiver abaixo desse limiar considera-se a não ocorrência do impulso de corrente, mas caso esteja acima, uma variável contadora da largura de impulso é incrementada. No fim do período de PWM (considerado logo após a zona de descida de PWM) esta variável contadora indicará a largura do impulso medido no período de PWM anterior. De notar que a resolução de medição da largura de impulso é tanto maior quanto menor for a periodicidades das interrupções. No entanto não convém definir períodos muito curtos sob pena de não atribuir suficiente largura de banda de CPU para o programa principal ser executado completamente num período de PWM. A periodicidade de 200µs foi o melhor valor encontrado e permite executar no total 95 medições, ou 31 para cada servo, ao longo de um período de PWM excluindo a zona de descida de PWM, ou seja, ao longo de 19 ms.

O valor do limiar a considerar depende da amplitude do impulso, mas sabendo que este apenas varia com a tensão de alimentação, mantendo uma amplitude sempre superior a 1.0V utilizando baterias de 7.4V, definiu-se o limiar como 1.0V. Note que este limiar encontra-se acima da tensão DC pelo que é necessário em cada período de PWM medir a tensão mínima e utilizá-la para o período seguinte como sendo a tensão base dos impulsos.

Contudo este método possui uma desvantagem. Em situações de elevada exigência, em que a corrente consumida é próxima da máxima, a largura do impulso de corrente pode ocupar praticamente todo período de PWM inibindo a capacidade de leitura da posição e da largura do impulso, dado que o valor mínimo corresponde ao topo do impulso de corrente. É, por isso, de evitar que esta situação se alcance, quer impondo limites físicos, quer pela adopção de estratégias de controlo que minimizem a corrente a consumir – uma delas é a limitação de velocidade.

3.2.3.2. Medição da Posição

Já foi referido anteriormente que a componente DC da saída do potenciómetro está relacionada à posição do servo, pelo que a tarefa de guardar o valor mínimo deste sinal, executada para efeitos de detecção do impulso de corrente, é aproveitada para o cálculo da posição angular do servo.
Convencionando a tensão mínima como correspondendo à posição +90º e a tensão máxima como -90º, recorrendo a uma relação linear podemos determinar a posição com ADC_{res} igual ao resultado da ADC, m sendo a relação entre a variação de posição e a variação de ADC_{res} (declive) e b o valor referência quando ADC_{res} é nulo (ordenada na origem).

$$\boxed{pos = b - ADC_{res} * m} \qquad \text{(Equação 2)}$$

$$m = \frac{+90^{\circ} - (-90^{\circ})}{ADC(1.8\text{V}) - ADC(-1.8\text{V})} = \frac{180^{\circ}}{368 - 163} = 0.878$$

$$b = pos + m * ADC_{res} = 90^{\circ} + m * ADC(0.8\text{V}) = 90^{\circ} + 0.878 * 163 = 233$$

...com $ADC(x)$ sendo o valor convertido pela ADC correspondente à tensão x (V):

$$ADC(x) = \frac{x}{5\text{V}} * (2^{bits_{quant}} - 1) = x * \frac{2^{10} - 1}{5\text{V}} = x * \frac{1023}{5\text{V}}$$

Logo $\boxed{pos = 233 - ADC_{res} * 0.878}$

Esta conversão é implementada no PIC através da macro:

```
#define POS(volt,...)   (origin - (volt)*SLOPE/QUOCIENT)
```

... em que *origin* corresponde à ordenada na origem *b*, *SLOPE/QUOCIENT* é o declive *m*, e *volt* é o resultado da conversão da ADC (ADC_{res}). Note que o declive *m* é inferior à unidade daí a necessidade do formato *SLOPE/QUOCIENT* para poder utilizar apenas valores inteiros (*SLOPE<QUOCIENT*).

Contudo, não é possível garantir que a gama da tensão de saída esteja compreendida entre 0.8 e 1.8V para qualquer servomotor, podendo-se verificar variações até cerca de 0.2V. Tal é problemático para o cálculo da posição angular, pois estamos a considerar uma relação de correspondência constante quando tal não acontece entre dois actuadores. No entanto, note que esta diferença é mais notória para o parâmetro b do que para o m. O valor de *b* exige que os extremos de posição possuam exactamente os valores enunciados, mas para o valor de *m* só é exigido uma excursão entre extremos de 1.0V o que já é mais frequente acontecer. Este pequeno detalhe permite-nos assim a utilização de uma rotina de calibração simples de modo a acertar o parâmetro b de acordo com o servo a lidar.
Considerando o parâmetro m constante, o que é uma aproximação aceitável, sempre que o sistema é ligado, se assegurarmos que cada servo está numa posição conhecida à priori, é fácil calcular o valor de b através da Equação 2 e utilizá-lo nas medições consequentes.

A rotina de calibração é executada sempre que o sistema arranca e segue o seguinte algoritmo:

1. A primeira mensagem de actuação já chegou? Só passar para o passo 2, quando afirmativo;
2. Actuação sobre os servomotores de forma a cumprir a posição solicitada pela unidade *master*;
3. Esperar dois segundos para a cumprimento do movimento e estabilização do sinal de posição;
4. Amostragem de 25 medidas da saída do servo (25 períodos de PWM de duração);
5. Cálculo da média aritmética do valor medido;
6. A partir da seguinte equação (baseada na Equação 2) determinar o valor de *b* a partir da posição solicitada pelo master e do valor médio ADC_{res}:
   $b = pos_{master} + ADC_{media} * m$
7. Activação dos filtros de medição sensorial.

Desta forma, a medição da posição é adaptado a cada servomotor de uma forma personalizada com mínimos erros de cálculo (apenas dependem de *m*).

3.2.3.3. Organização das interrupções de medição sensorial

A medição de tensões analógicas a partir de uma ADC, como se sabe, não é imediata, demorando um determinado intervalo de tempo até o resultado estar pronto. Para se proceder à amostragem é necessário percorrer um conjunto de passos:

1. Selecção do entrada do multiplexer correspondente ao servo a ler;
2. Esperar que o multiplexer efectivamente seleccione a entrada e tenha uma saída estável (alguns microsegundos) – tempo de estabilização;
3. Esperar cerca de 20µs para a ADC possuir um valor estável na sua entrada – tempo de aquisição;
4. Iniciar conversão da ADC;
5. Esperar que a ADC termine a conversão (cerca de 40µs) – tempo de conversão;
6. **Leitura do valor convertido e processamento (cerca de 10µs);**

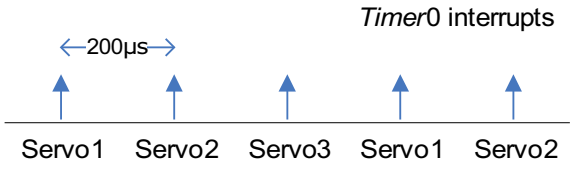

Fig. 42: Multiplexagem na leitura dos servos.

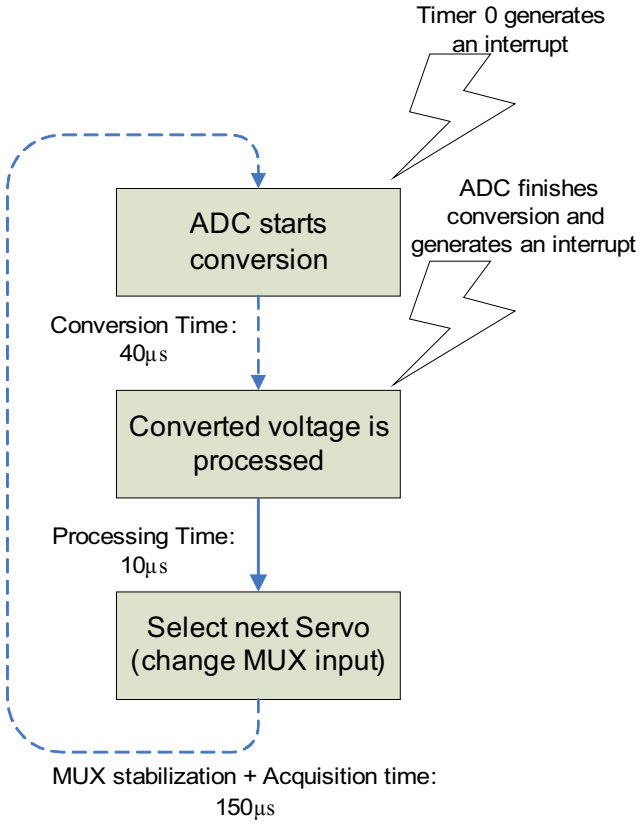

Fig. 41: Algoritmo de leitura dos três servomotores.

Só depois de executados os primeiros 5 passos, a tensão de saída do servo pode ser medida com segurança. Somando estes períodos de tempo, incluindo o tempo de execução do código de processamento do resultado, verificamos que todo o processo é executado em menos de 80µs, pelo que restam 120µs para execução de outras tarefas iniciadas pela função *main*.
No entanto, não nos esqueçamos que temos aplicar este processo a três actuadores. Para não sobrecarregar o CPU de repetir este procedimento três vezes em cada período de 200µs optou-se por efectuar multiplexagem na leitura, ou seja, em cada período de 200µs apenas um servo é lido, mas no período seguinte o servo a ler é outro, completando-se a leitura dos três servos ao fim de três períodos (600µs). Logo após a leitura do terceiro, volta ao primeiro repetindo de novo todo o processo (Fig. 42).

A Fig. 41 apresenta o algoritmo implementado, com uma gestão temporal mais eficiente, seleccionando a entrada do *multiplexer* correspondente ao servo seguinte no fim do código que processa as interrupções provenientes do *timer* 0 (RSI), e não no início, deixando livres 150µs entre o fim da execução do código e o atendimento da próxima interrupção para estabilização da saída do *multiplexer* e o cumprimento do tempo de aquisição da ADC. Desta forma, quando o *timer* 0 voltar a gerar uma interrupção poder-se-á arrancar com a ADC de imediato apenas tendo de esperar por uma interrupção por parte da ADC indicando a prontidão do valor convertido. Em termos de tempo efectivo envolvido na leitura apenas temos os 10µs do código de processamento do valor convertido, uma vez que sempre que é necessário introduzir um tempo de espera o CPU não fica bloqueado em modo de espera, mas retorna ao programa normal deixando encarregues às interrupções a tarefa de voltar prosseguir a sequência de operações.

Um outro aspecto a salientar é a coabitação entre as interrupções de medição sensorial e as interrupções para actuação (Fig. 35 e Fig. 40). Dada a igualdade, em termos de prioridade entre os dois tipos, um mecanismo que evite a interferência entre os dois torna-se fundamental para evitar atrasos que comprometam a

fiabilidade de funcionamento destes dispositivos. Um atraso na interrupção para a gestão da actuação poderia resultar num sinal de PWM modificado que alteraria sem intenção a posição do servo, ou um atraso nas interrupções envolvidas na medição poderia comprometer a validade dos resultados. Para evitar problemas deste género impuseram-se as seguintes condições:

- As interrupções gestoras do sinal de PWM nunca devem sofrer interferência por parte de outras e devem ser sempre cumpridas no instante a que foram programadas;
- As interrupções gestoras da medição sensorial nunca podem ser interrompidas a meio da sua execução, ou seja, quando a primeira operação é executada – arranque da ADC – é fundamental a sua execução sem atrasos até ao fim. Deste modo o processo de leitura sensorial deve ser considerado como uma operação atómica. Contudo este processo, como um todo, pode ser inibido de modo a não interferir com as interrupções gestoras do PWM.

O mecanismo de coabitação deve então seguir este procedimento: em cada interrupção proveniente do *timer* 0 (medição sensorial) ainda antes de se proceder à primeira operação – o arranque da ADC – deve-se verificar se há tempo para a execução completa de todo o processo de medição ainda antes da próxima interrupção gestora do PWM:

- ➔ Antes da interrupção do *timer* 1 responsável por elevar o sinal de PWM a 1;
- ➔ Antes da primeira interrupção do *timer* 2 que inicia a zona de descida do PWM a 0.

O intervalo de tempo mínimo considerado, para o processo de medição sensorial estar autorizado a iniciar deve corresponder à periodicidade do *timer* 0, ou seja, 200µs. Se para a próxima interrupção de actuação faltar mais de 200µs a medição sensorial é autorizada a arrancar, pois é garantido que finaliza antes dela chegar. Quando a zona de descida de PWM começa, as interrupções de medição sensorial devem ser inibidas de modo a poder dedicar toda a largura de banda do CPU para as interrupções de alta frequência. Quando esta zona finaliza os dados sensoriais amostrados durante o período de PWM anterior são tratados e estas interrupções são reactivadas.

Quanto ao processamento do sinal de saída do servo, a Fig. 44 descreve o algoritmo utilizado para cada servo em cada iteração ao longo de um período de PWM (caixa "*converted voltage is processed*" da Fig. 41).

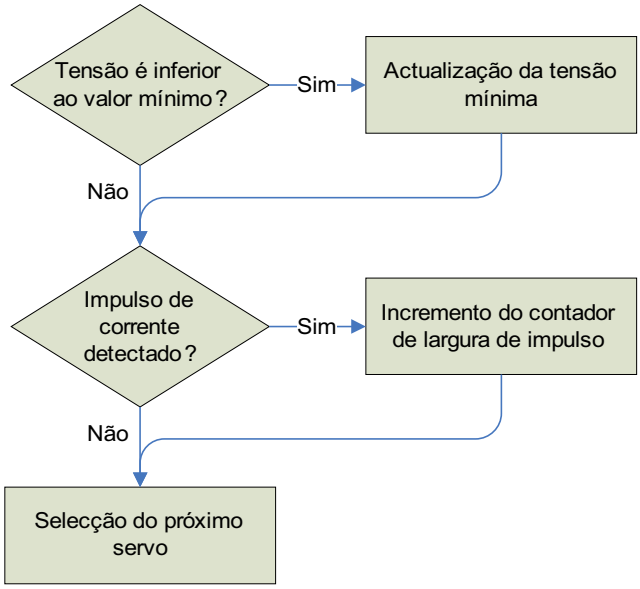

Fig. 44: Algoritmo de processamento da tensão medida para cada servomotor.

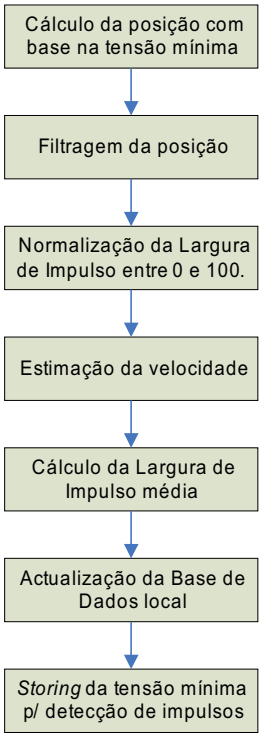

Fig. 43: Processamento final (fim do período de PWM).

No fim de cada período de PWM, os valores da tensão mínima e da largura de impulso são processados em cada servo para determinação da posição, velocidade média e corrente consumida (Fig. 43):

1. Cálculo da posição com base na tensão mínima: A posição correspondente ao valor mínimo da ADC é calculado a partir da relação linear...

$$pos = b - ADC_{min} * m$$

2. Filtragem da posição: O valor da posição obtido é filtrado de modo a evitar variações bruscas. São utilizados dois filtros:

   a) Filtro de média não linear: este filtro, muito usado em processamento de imagem, é usado apenas para remover picos nos valores medidos, sem interferir de de nenhuma forma no sinal na ausência deles. O atraso introduzido é apenas de uma amostra.

   y → amostra obtida
   y_max → máximo valor da amostra admitido
   y_min → mínimo valor da amostra admitido
   y_new → Valor filtrado
   y_prev → Valor da amostra anterior (não filtrado)

```
// Limitação da saída
if (y>y_max)        y_new=y_max;
else if (y<y_min)   y_new=y_min;
else                y_new=y;

// Actualização das amplitudes limite
if (y>y_prev)  {
    y_max=y;
    y_min=y_prev;
}
else  {
    y_max=y_prev;
    y_min=y;
}

// Actualização da saída anterior
y_prev=y;
```

   b) Filtro de média linear: este filtro é aplicado a seguir ao não linear e tem como função a suavização do sinal resultante. Apenas é feita uma média igualitária entre a nova amostra e a anterior.

   ynew_prev → Valor filtrado anterior

```
// Filtro Passa-baixo
y_new = (y_new+ynew_prev)/2;
ynew_prev = y_new;
```

3. Normalização da largura do impulso de corrente: De modo a permitir a redefinição da periodicidade do timer 0 ($timer0_{per}$) sem interferir no resultado da largura do impulso de corrente, o resultado é normalizado para a gama entre 0 e 100 (percentagem) através de uma regra proporcional:

$$Largura_{norm} = Largura * \frac{100}{19\text{ms} / (timer0_{per} * 3\, servos)}$$

4. Estimação da velocidade: Embora não seja possível medir directamente a velocidade, é possível estimá-la a partir da variação de posição. Definindo a velocidade mínima mensurável como sendo 10º/s precisamos de medir a variação de posição correspondente a 100ms uma vez que o valor mais pequeno que conseguimos medir é 1º. No entanto, é desejável que tenhamos sempre a velocidade disponível todos os 20 ms sem ter que esperar 100 ms para poder ler este dado. Tal é possível usando arrays circulares (Fig. 45):

   Um array de 5 posições pode armazenar até 5 posições antigas – *pos*(*n*-1) até *pos*(*n*-5) – em que a

posição mais antiga corresponde à posição de há 100 ms atrás (20ms*5). Se em cada período de PWM armazenarmos a posição acabada de medir no elemento mais antigo, em todos os 20 ms temos disponível o valor de posição de há 100ms atrás, ainda antes da actualização do array com o novo valor. Desta forma, a cada 20ms, podemos calcular a variação de posição verificado nos últimos 100 ms.

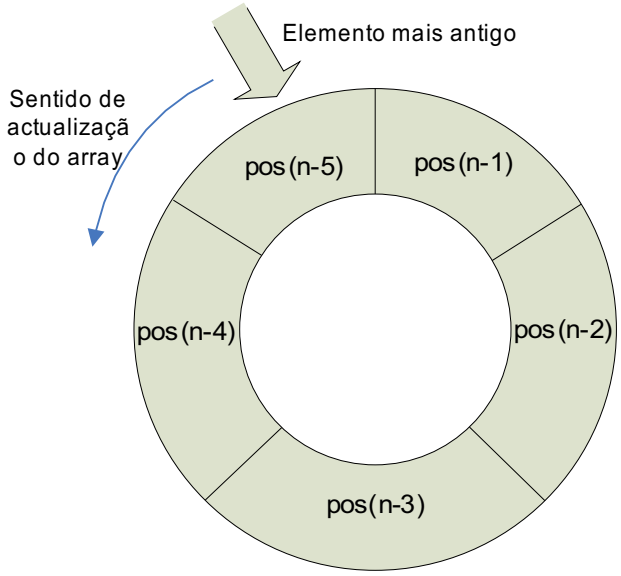

Fig. 45: Array circular para armazenamento de posições.

Este array tem o nome de circular, pois o elemento mais antigo, a que chamaremos de *index*, está sempre em contínua rotação ao longo do array: quando introduzimos um valor no elemento mais antigo, esse elemento passa a ser o mais novo e o elemento a seguir passa a ser o mais antigo.

a) Estimação da velocidade: $vel = pos(n) - pos(n-5) = pos_{medido} - array[index]$
b) Actualização do array: $array[index] = pos_{medido}$
c) Actualização do elemento mais antigo $index = remain[(index+1),5]$

5. Cálculo da largura de impulso média: Dada a instabilidade do valor de corrente medido (largura de impulso) implementou-se o cálculo da média de todos os valores de corrente medidos nos últimos 100ms. De modo a ter sempre um valor actualizado todos os 20ms seguiu-se a mesma estratégia que para a estimação de velocidade: um array circular para armazenamento dos cinco valores de corrente mais recentes. A cada 20ms é introduzido o novo valor medido e é calculada a média do conjunto considerando esse valor como a final.

$$Corrente_{media} = \frac{\sum_{index=0}^{4} array[index]}{5}$$

6. Actualização da base de dados local (sensorial): Ver secção 1.2.4.1 (base de dados local).

7. Armazenamento da tensão mínimo medida: necessário para o conhecimento da tensão base dos impulsos de corrente para o período de PWM seguinte.

3.3. ESTUDO DO SERVOMOTOR EM MALHA ABERTA

Agora que os detalhes de funcionamento do microcontrolador foram apresentados, vamos agora estudar o comportamento do servomotor sob determinadas condições de modo a avaliar a sua performance. O estudo será feito do ponto de vista de um sistema no qual aplicaremos uma entrada e queremos saber qual é a sua resposta (Fig. 46).

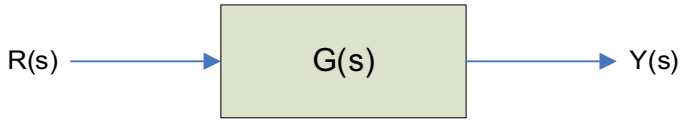

Fig. 46: Representação do servomotor por uma função de transferência G(s).

Ora, tendo sido discutido o funcionamento interno do servomotor na secção 2.1 (Fig. 29), infelizmente apenas era uma aproximação ao nosso modelo e além disso não conhecemos os valores das grandezas enunciadas. Por tudo isto iremos estudar o comportamento do sistema G(s), desconhecendo a sua função de transferência.

Embora, pudéssemos tentar encontrar esta expressão experimentalmente através de métodos bastante conhecidos como é o caso das regras de Ziegler-Nichols, tal não nos leva muito longe, pois como é sabido, as juntas da plataforma humanóide estão sujeitas a variações de inércia pelo que o comportamento dos servos torna-se bastante não linear. Teríamos, por isso, uma função de transferência diferente para cada carga aplicada.

Este capítulo tem como objectivo estudar primeiramente a resposta em malha aberta do servo, ou seja, aplicando um sinal (ou uma sequência de sinais) de PWM na entrada r(t), analisar a resposta pela leitura da posição angular y(t). Rotinas em MatLab para uso pela unidade principal foram especialmente construídas para este efeito enviando para o *slave* respectivo uma ordem de actuação, e logo de seguida monitorar a posição do servo o tempo suficiente para permitir captar informação informação importante.

Algum vocabulário não muito comum será utilizado pelo que convém, antes de tudo, esclarecer o seu significado para evitar qualquer dúvida ou ambiguidade:

- ➔ Tempo de crescimento/subida: tempo que a resposta demora a crescer entre 10% e 90% da distância total a percorrer;
- ➔ Tempo de atraso: instante que a resposta atinge 50% da distância total percorrida;
- ➔ Tempo de pico: instante em que a resposta passa pela posição máxima/mínima (depende do sentido de deslocamento);
- ➔ Tempo de estabelecimento: tempo necessário para que a resposta entre, sem voltar a sair , numa determinada vizinhança , previamente especificada , do valor final da resposta. Dois a cinco por cento é normalmente a margem especificada.
- ➔ Overshoot: oscilação verificada em torno do valor final no fim da resposta. O seu valor corresponde normalmente à relação entre a margem máxima de oscilação e a distância total percorrida.
- ➔ Erro em regime estacionário: diferença entre o valor final e o valor desejado após estabilização da resposta.

Estas definições correspondem a características da curva de resposta do sistema a testar e normalmente são usados para avaliar a sua performance (Fig. 47). No caso de um actuador ideal, todos estas características deviam ser nulas, mas infelizmente tal não existe na realidade: os sistemas levam tempo a reagir e a atingir o seu valor final, e por vezes podem entrar em oscilação (*overshoot*) quando a entrada é demasiado exigente. Este capítulo procurará perceber o quanto os servomotores se desviam da resposta ideal, e posteriormente tentaremos encontrar soluções para a sua melhoria.

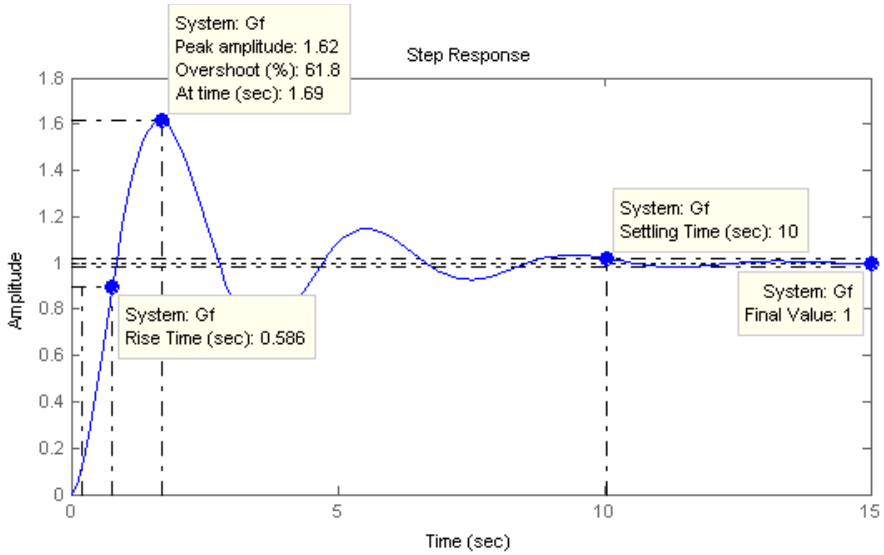

Fig. 47: Exemplo da resposta de um sistema (Gf) com a indicação das suas características.

3.3.1. Resposta ao Degrau em Malha Aberta

Com uma carga aplicada no eixo, segundo a forma apresentada na Fig. 32, inicializou-se o servo na posição inicial de -45º e enviou-se um comando de actuação para a posição de +45º – um degrau é aplicado. Monitorizou-se o percurso percorrido através do comando de leitura de posição durante 1.5s para duas cargas diferentes: 258 e 1138g (Fig. 48).

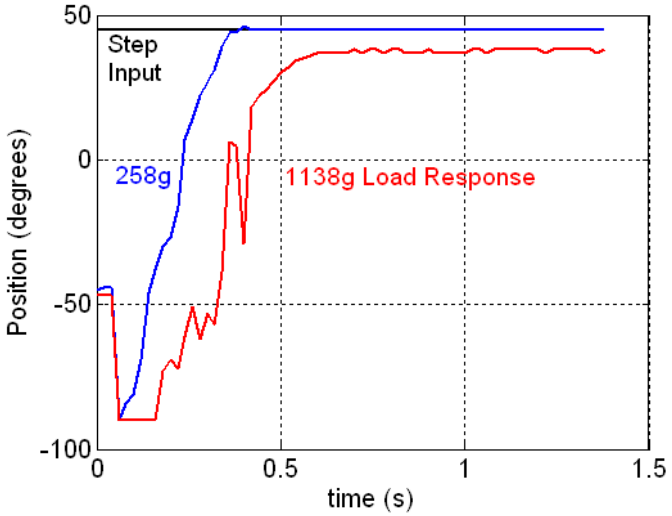

Fig. 48: Comparação das respostas ao degrau para duas cargas no percurso de -45 para +45º.

Requested position º	measured position º	Error º	Torque (Nm)
-80	-80	0	0.198
-60	-62	2	0.569
-40	-45	5	0.872
-20	-28	8	1.069
0	-9	9	1.138
+20	+11	9	1.069
+40	+33	7	0.872
+60	+55	5	0.569
+80	+80	0	0.197

Tabela 31: Erros em regime estacionário em diferentes posições para uma carga de 1138g.

Comparando as respostas da carga de 258g com a de 1138g, constata-se uma diferença dos tempos de subida evidenciando um atraso maior para cargas elevadas, o que é compreensível uma vez que o esforço dispendido é maior. Um segundo aspecto é a diferença no erro em regime estacionário: para a massa leve o erro é praticamente nulo, mas para a mais pesada já é mensurável um erro de cerca de 9º.

Adicionalmente, fez-se uma experiência tendo em vista estudar a relação do erro em regime estacionário com o binário resultante da força gravítica. Deslocando o servo para um conjunto de posições conhecidas, para cada uma delas, esperou-se pela finalização do movimento e pela estabilização do sinal de posição, anotando de seguida a posição medida pelo microcontrolador (Tabela 31). Comparando a posição solicitada com a efectiva observa-se que o erro aumenta à medida que a posição se aproxima do ponto 0º, o que

demonstra que quanto maior é o binário gravítico maior é a dificuldade em atingir a posição final resultando num erro em regime estacionário não nulo.

Outro aspecto a salientar é a visível instabilidade durante a realização do trajecto o que é mais notório para a carga elevada do que para a baixa. Além disso observa-se um salto no início da trajectória para posições inferiores a -45º seguidamente com inversão de velocidade em direcção ao valor final. No entanto este comportamento não foi o que aconteceu realmente, tendo-se observado um movimento rápido e sem oscilações durante todo o percurso. Tal sugere que a posição lida a partir do potenciómetro está a ser perturbada por algo.

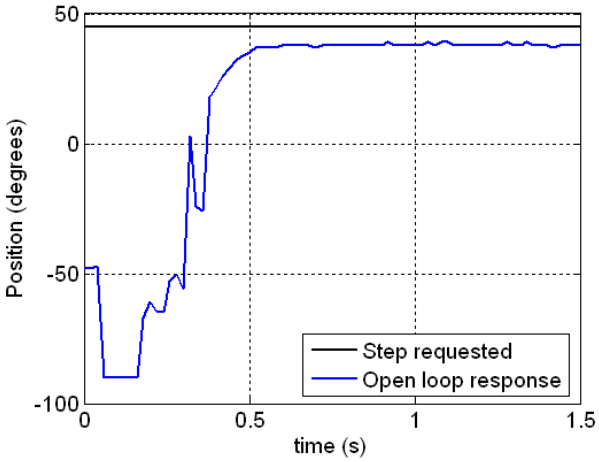

Fig. 49: Resposta ao degrau de -45 para +45º com uma carga de 924g.

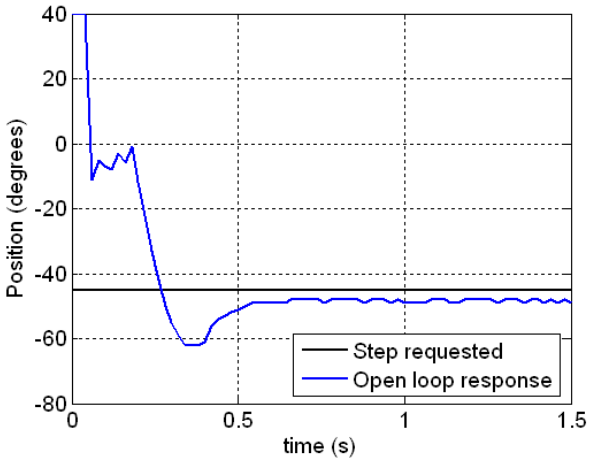

Fig. 50: Resposta ao degrau de +45 para -45º com uma carga de 924g.

Os gráficos da Fig. 49 e Fig. 50 apresentam a execução do mesmo percurso mas nos dois sentidos para uma carga de 924g. Como se pode observar, continuam a surgir os picos de corrente no trajecto inicial em ambos os casos. No caso da descida de +45 para -45º verifica-se um abaixamento brusco na posição medida até metade do trajecto onde ocorre uma recuperação.

A presença de acelerações bruscas, como acontece no arranque da trajectória, pode provocar picos de corrente, que em termos sensoriais, correspondem a um impulsos de corrente que podem ocupar toda o período de PWM. Nestas circunstâncias a posição considerada como sendo a tensão mínima deixa de poser ser medida sofrendo um aumento em tensão para o topo do impulso, que, em termos de posição angular, corresponde a uma descida brusca, tal como os gráficos nos mostram.
Neste sentido, pode-se dizer que a presença do impulso de corrente está a mascarar as medições de posição do servo agravando os resultados na presença de cargas elevadas. No entanto, tal deixa de se verificar nos últimos instantes ainda antes da finalização da trajectória e durante a fase estacionária, demonstrando que não só a massa da carga aumenta as exigências de corrente, como também a velocidade e as acelerações bruscas.

3.3.2. Controlo de Velocidade

Como se pode concluir, os servomotores são muito sensíveis a velocidades e acelerações bruscas provocando muito facilmente picos de corrente que impedem a correcta leitura da sua posição. De modo a prevenir este efeito, tentou-se introduzir algum controlo de velocidade de modo a eliminar estas variações bruscas, com a adicionante de ganharmos na capacidade de poder regular a velocidade segundo as nossas necessidades e de possibilitar movimentos mais suaves. No entanto, como não temos possibilidade de fazer o controlo directo de velocidade, utilizar-se-á o controlo de posição para executar trajectórias que no seu todo definem uma velocidade média que pode ser configurável.

Até agora temos vindo a aplicar degraus de posição aos servos tal como exemplifica a Fig. 51. Se aplicarmos uma sucessão de degraus de variação de amplitude e intervalo de tempo o mais pequenos possíveis, cuja

amplitude final de cada degrau aumenta proporcionalmente até atingir a posição desejada, temos a aplicação de uma rampa de posições cujos extremos de posição e duração total definem a velocidade média do movimento. A Fig. 52 apresenta um exemplo de uma trajectória em rampa desde -45º até +45º com uma duração de 1.8s, o que corresponde a uma velocidade média de 50º/s.

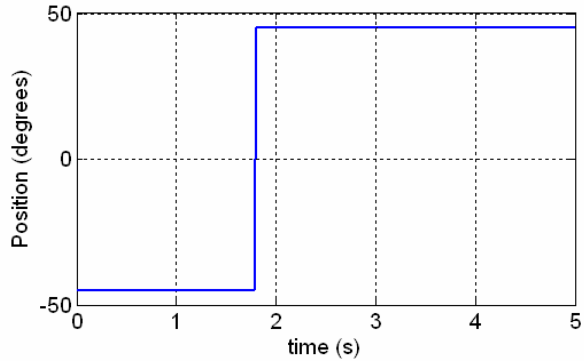

Fig. 51: Aplicação de um degrau de -45º para +45º no instante t=1,8s.

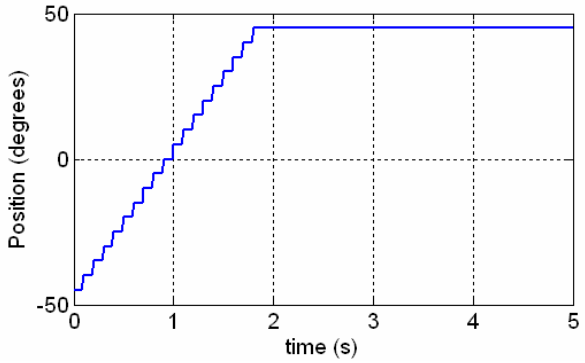

Fig. 52: Aplicação de uma rampa de posição de velocidade média 50º/s.

Enquanto que a aplicação de um degrau introduz um delta de Dirac na velocidade e na aceleração provocando facilmente picos de corrente, a trajectória em forma de rampa contêm essas variações bruscas pela aplicação sucessiva de pequenos degraus o que limita a velocidade do servo e também a sua necessidade de consumo de corrente. Esta trajectória pode ser implementada através da Equação 3.

$$pos = pos_0 + v_{med} \cdot t = pos_0 + v_{med} \cdot n \cdot Ta \qquad \text{com } T_a\text{=20ms (período de PWM).} \qquad \text{(Equação 3)}$$
$$vel = v_{med} \qquad acel = 0$$

No entanto, se quisermos optimizar ainda mais o consumo de corrente, podemos implementar trajectórias, de modo a limitar os deltas de Dirac de aceleração que continuam a verificar-se no caso da rampa. Se além de variarmos a posição, também variarmos a velocidade de modo a ser nula no início e no fim da trajectória, a necessidade de consumo de corrente decai ainda mais. Tal é exequível através de uma equação polinomial de terceira ordem que introduz velocidade zero no início e no fim de cada trajectória.

$$pos = c_0 + c_1 \cdot t + c_2 \cdot t^2 + c_3 \cdot t^3 \qquad vel = c_1 + 2 \cdot c_2 \cdot t + 3 \cdot c_3 \cdot t^2 \qquad acel = 2 \cdot c_2 + 6 \cdot c_3 \cdot t$$

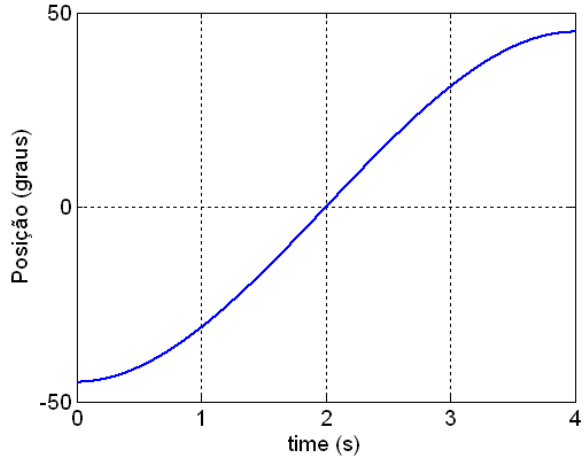

Fig. 53: Trajectória polinomial de terceira ordem.

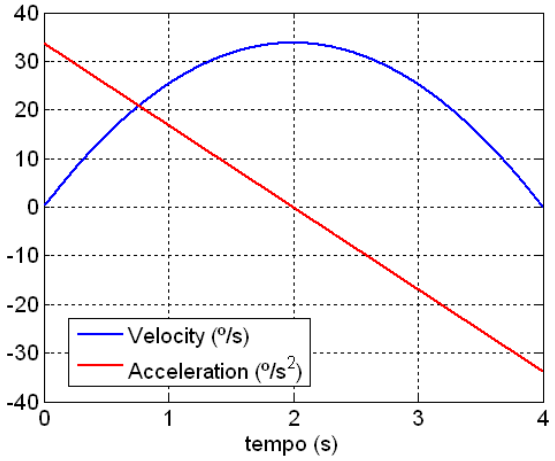

Fig. 54: Comportamento da velocidade e da aceleração na trajectória polinomial.

Como se pode verificar, deixa de se verificar deltas de Dirac até à segunda derivação da posição (aceleração). Se quiséssemos ir ainda mais longe, podíamos aumentar a ordem do polinómio de modo a garantir aceleração nula no início e no fim. No entanto tal não foi implementado dado ao facto de que quanto maior é a ordem do polinómio maior é a velocidade instantânea a meio do trajecto. Ordem três corresponde a um bom compromisso para o que é preciso.

3.3.2.1. Respostas em malha aberta

Voltando a amostrar a resposta dos servos em malha aberta, agora com a implementação de trajectórias, podemos observar, para o caso da rampa (Fig. 55 e Fig. 56), a estabilidade acrescida nas respostas qualquer que seja a carga aplicada. Os efeitos de picos de posição e as oscilações durante o percurso praticamente desapareceram demonstrando o consumo controlado de corrente com esta solução.

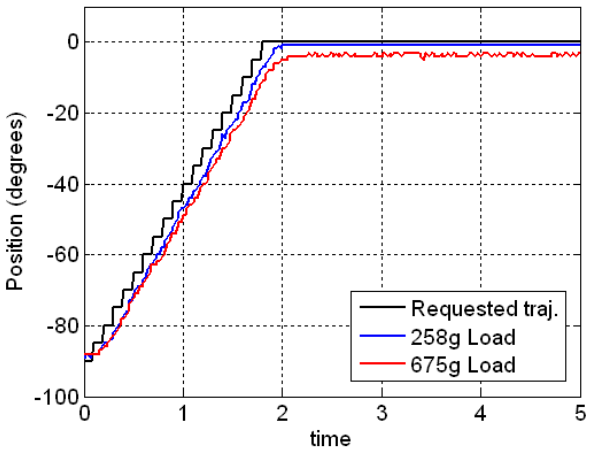

Fig. 55: Resposta à rampa com duas cargas diferentes (Δp=5º, Δt=100ms).

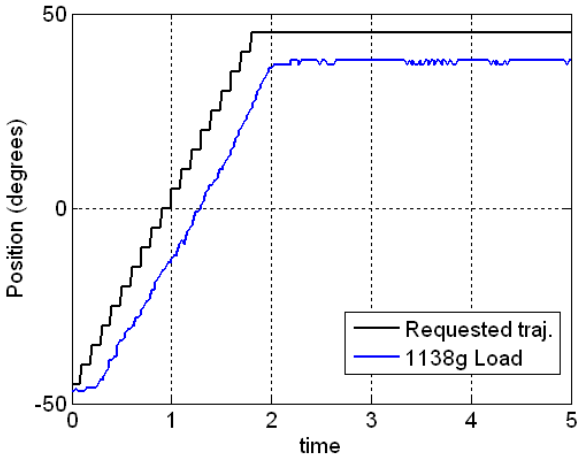

Fig. 56: Resposta à rampa com uma carga pesada (Δp=5º, Δt=100ms).

A Fig. 57 apresenta as respostas correspondentes às trajectórias polinomiais com a observação dos mesmos resultados que com a rampa.

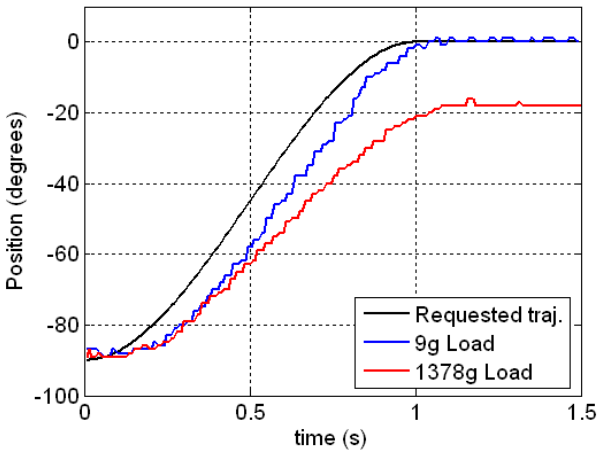

Fig. 57: Resposta ao polinómio para duas cargas diferentes (T_{traj}=1s).

No entanto, continua-se a verificar o crescente tempo de atraso e erro em regime estacionário com a carga aplicada em relação à trajectória esperada. Para corrigir estas características torna-se importante realizar alguma espécie de controlo adicional ou ao nível do controlador interno do servo, ou externamente, usando o sinal de posição como *feedback* e o sinal de PWM para controlo da posição.

3.4. ESTUDO DO SERVOMOTOR EM MALHA FECHADA

Este capítulo descreve algumas estratégias para melhoria da resposta dos servos, em termos de:

- ➔ tempo de subida;
- ➔ tempo de estabelecimento;
- ➔ erro em regime estacionário.

O objectivo será minimizar ao máximo estes parâmetros de modo a aproximar ao máximo a resposta da trajectória solicitada.
Várias metodologias podem ser seguidas para compensação, como por exemplo a substituição da electrónica de controlo dos actuadores. No entanto, procura-se por métodos de compensação que não modifiquem estas unidades de forma a permitir a fácil substituição em caso de necessidade. Por razões de simplicidade realizar-se-á o controlo externamente ao servo usando para isso o microcontrolador para implementar a lei de controlo mais adequada. Com a implementação do controlador por software, é possível alterar os parâmetros ou a estrutura do controlador ou modificando simplesmente o código, ou por troca de informação do PC para o slave respectivo, evitando assim intervenções ao nível do hardware.

3.4.1. O Controlador

A Fig. 58 descreve a metodologia a usar: o controlador representado pelo bloco $G_C(z)$ é implementado ao nível do microcontrolador que fará uso do sinal de PWM $u(t)$ e do sinal de posição $p(t)$ como *feedback* para comparação com a posição desejada $r(t)$.

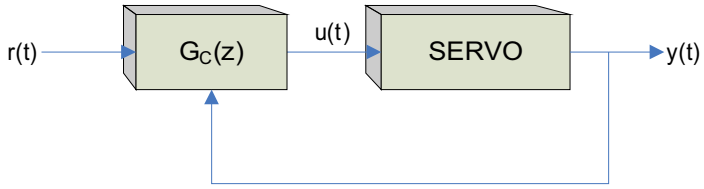

Fig. 58: Controlo externo do servomotor.

A partir daqui já não se dirá que um determinado duty-cycle do PWM corresponde a uma determinada posição, pois tal depende muito da inércia aplicada sobre o eixo, mas simplesmente é um sinal de controlo que para uma determinada carga corresponde a uma determinada posição. Para tal, a lei de controlo $G_C(z)$ seguirá um método para procurar o sinal de PWM mais adequado para que o sinal de posição medido p(t) coincida com a posição solicitada r(t). Esse método é apresentado na e é baseado num compensador clássico do tipo PID.

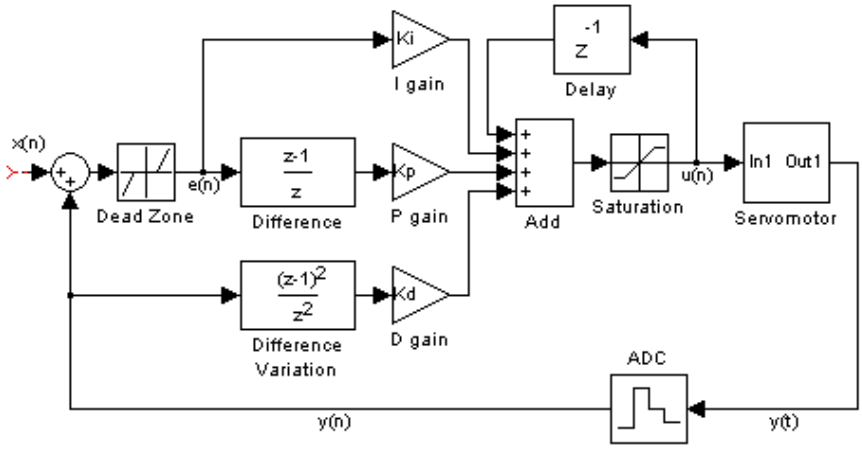

Fig. 59: Compensador PID incremental.

Este compensador é baseada na equação de compensação PI-D no domínio de Laplace, sendo convenientemente adaptada para o caso digital:

$$U(s)=\left[k_i\cdot\frac{1}{s}+k_p+s\cdot k_d\right]\cdot E(s)$$

No domínio digital (z) temos:

$$U(z)=\left[K_I\cdot\frac{1}{1-z^{-1}}+K_P+K_D\cdot(1-z^{-1})\right]\cdot E(z),\quad com\ \ K_I=k_i\cdot T_S,\ \ K_P=k_p,\ \ K_D=\frac{k_d}{T_S}\ \ (T_S=20\text{ms})$$

Colocando a equação na forma incremental:

$$U(z)\cdot(1-z^{-1})=K_I\cdot E(z)+K_P\cdot(1-z^{-1})\cdot E(z)+K_D\cdot(1-z^{-1})^2\cdot E(z)$$

Em termos numéricos:

$$\Delta u(n)=K_I\cdot e(n)+K_P\cdot\Delta e(n)+K_D\cdot\Delta\Delta e(n)$$

... em que Δ corresponde a uma variação relativamente à amostra anterior.

$$u(n)-u(n-1)=K_I\cdot e(n)+K_P\cdot[e(n)-e(n-1)]+K_D\cdot[(e(n)-e(n-1))-(e(n-1)-e(n-2))]$$

$$u(n)=u(n-1)+K_I\cdot e(n)+K_P\cdot[e(n)-e(n-1)]+K_D\cdot[e(n)-2\cdot e(n-1)+e(n-2)]$$

De modo a minimizar a instabilidade resultante do aumento da ordem da equação diferencial substituamos na secção derivativa $e(n)$ por $r(n)$-$y(n)$ com $r(t)$ constante. Desta forma a instabilidade da componente derivativa deixa de estar relacionado com as variações do sinal de referência[3].

$$\boxed{u(n)=u(n-1)+K_I\cdot e(n)+K_P\cdot[e(n)-e(n-1)]-K_D\cdot[y(n)-2\cdot y(n-1)+y(n-2)]}$$

Esta é a lei de controlo implementada no microcontrolador cujos parâmetros K_I, K_P e K_D são passados pelo PC através da rede de comunicações. Prevê-se que a componente integral resolva o problema do erro em regime estacionário e as restantes componentes lidem com a velocidade do sistema, mas que fique bem claro, que não é lícito importar "ideias feitas" provenientes de disciplinas relacionadas com o controlo de sistemas, uma vez que nem sequer iremos lidar com entradas em degrau, mas sim numa sucessão de degraus, o que pode resultar em efeitos diferentes no resultado final.

Note que o controlo é realizado de forma incremental, calculando em cada iteração o incremento a dar ao sinal de controlo $u(t)$ a fornecer ao servo. Optou-se por esta solução dadas as vantagens que oferece:

- ✔ Não são necessárias variáveis de elevada resolução para armazenar o resultado de somas;
- ✔ Protecção *wind-up*;
- ✔ Transferência *bumpless* simplificada.

A alternativa ao algoritmo incremental exigiria a actualização de um somatório a cada iteração (elemento $1/s$) o que implicaria o recurso a variáveis de elevada dimensão (*longs* ou *doubles*) para armazenar o resultado, o que nem sempre é favorável em arquitecturas baseadas em microcontroladores.

Além disso poderia ocorrer a requisição de uma posição fora dos extremos do servo (-90 e +90º) sem que a integração seja capaz de inverter a tendência do sinal de controlo uma vez que o seu incremento apenas se limita ao valor do extremo. Só ao fim de algum tempo, que não seria pouco, a soma pode ser suficiente para inverter a tendência, resultando numa perda significativa na reactividade da resposta. Este fenómeno denomina-se por *wind-up* e é devida à saturação do actuador. Embora haja bastantes soluções para este

3 Recomenda-se o teste da lei de controlo sem a inclusão do sinal de referência também na componente proporcional:
$u(n)=u(n-1)+K_I\cdot e(n)-K_P\cdot[y(n)-y(n-1)]-K_D\cdot[y(n)-2\cdot y(n-1)+y(n-2)]$

problema, a mais simples é a do algoritmo incremental, pois não existe qualquer integrador, fazendo com que numa situação de saturação o sinal de controlo $u(n)$ deixe automaticamente de aumentar resultando imediatamente na inversão da tendência.
Um outro pormenor são as transferências *bumpless*, que não são nada mais do que a activação e desactivação do controlador em pleno funcionamento do servo. Pretende-se que quando o controlador é ligado ou desligado, o actuador não sofra qualquer variação brusca de posição. Embora não haja nenhum problema, por parte das duas soluções, na situação de desactivação do controlador, o problema surge na reactivação. Na solução com integrador, se a integração parar de funcionar durante a desactivação, quando reactivado, o sinal de controlo $u(n)$ não corresponderá ao sinal $r(t)$ que anteriormente era aplicado directamente, pois o resultado da integração deixou de ser actualizado, o que resulta num deslocamento brusco para uma posição desconhecida.
Por outro lado, se nunca se parar a integração corremos o risco da soma atingir valores excessivamente elevados ou mesmo de sofrer *overflow*, bastando para isso que os pedidos de actuação nunca correspondam ao valor de *feedback* – tal é frequente na presença de cargas. O resultado reflectir-se-ia em movimentos bastante oscilatórios na reactivação, acabando por provocar o fenómeno de *wind-up*!
Uma solução seria, em todas as reactivações, calcular o valor da soma de modo a que o sinal de controlo $u(t)$ correspondesse ao valor de $r(t)$ e inicializar a soma com esse valor. No caso do algoritmo incremental, se definirmos o valor $u(n\text{-}1)$ como o último valor aplicado no servo, quer com o controlador ligado, quer desligado, escusamo-nos de qualquer preocupação com este procedimento.

No entanto há uma desvantagem com este algoritmo, resultante do aumento da ordem de um para dois na remoção do integrador. Na presença de ruído, este compensador torna-se mais sensível podendo levar o sistema mais facilmente à instabilidade. Daí a necessidade do formato PI-D em que é o sinal de saída e não o de erro o utilizado na componente derivativa eliminando qualquer possibilidade de instabilidade resultante de variações do sinal referência $r(t)$. Mesmo assim recomenda-se o uso de valores baixos para o parâmetro K_D.

3.4.2. Controlo Integral (I)

Vamos agora testar o controlador para várias cargas e percursos utilizando as trajectórias em forma de rampa e de polinómio de terceira ordem, usando como referência as respostas em malha aberta para a sua avaliação.

Comecemos por utilizar a componente integral, definindo os parâmetros K_P e K_D a zero. Experimentando o valor de 0.08 para K_I para duas massas de elevado valor (Fig. 60 e Fig. 61) pode-se observar em ambos os casos a eliminação do erro em regime estacionário. No caso da Fig. 61 a diferença de dois graus é devida ao efeito da banda morta presente na entrada do controlador. Repare no sinal de saída do controlador solicitando ao servo uma posição mais elevada do que a desejada de modo a que ela seja cumprida na presença da carga eliminando assim o erro em regime estacionário.

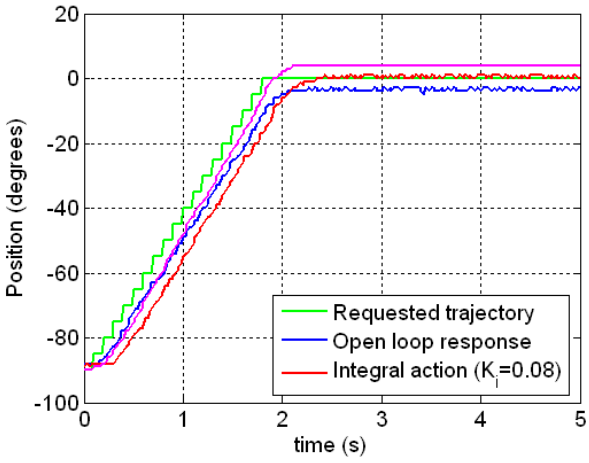

Fig. 60: Resposta à rampa com uma carga de 675g (K_I=0.08).

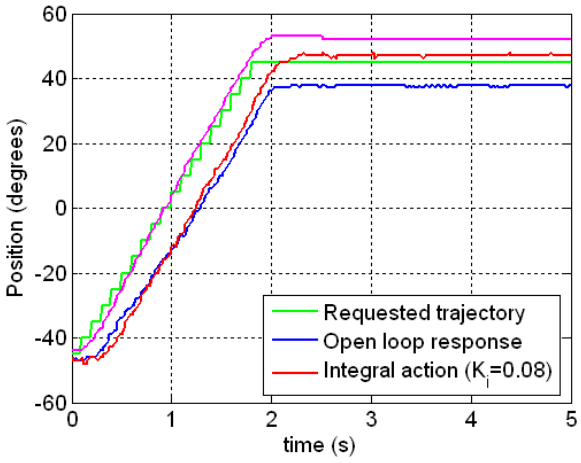

Fig. 61: Resposta à rampa com uma carga de 1129g (K_I=0.08).

Aumentando o valor de K_I para 0.20 e realizando o trajecto de -45 para +45º com uma carga de 1129g (Fig.

62 e Fig. 63), além da ausência do erro em regime estacionário, o tempo de atraso da resposta relativamente à trajectória solicitada é melhorado na presença do controlador o que beneficia o tempo de estabelecimento.

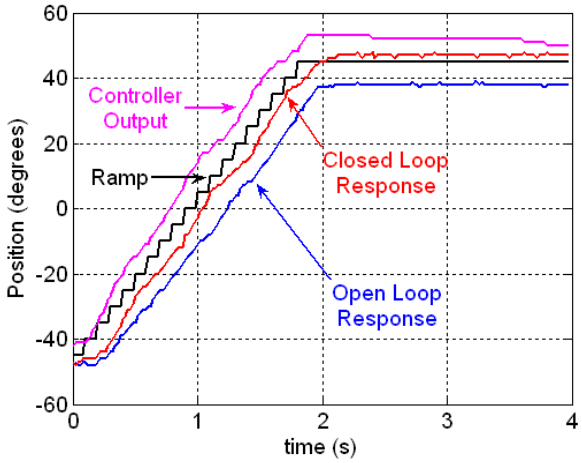

Fig. 62: Resposta à rampa com uma carga de 1129g (K_I=0.20).

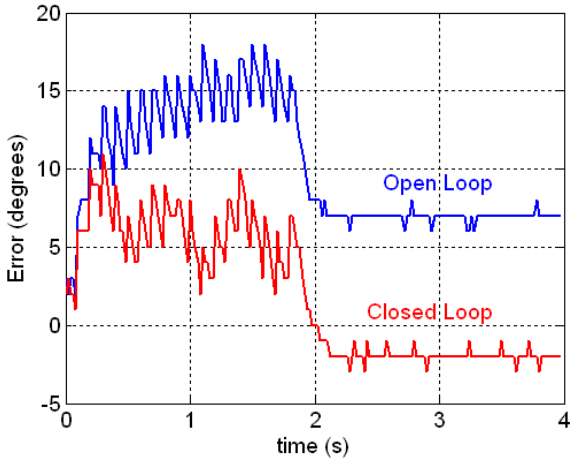

Fig. 63: Erro da resposta da Fig. 51.

De modo a estudar os efeitos do parâmetros K_I na resposta do servo, realizou-se uma experiência, no qual para uma massa elevada de 924g efectuando um percurso fixo (-45 a +45º) experimentaram-se vários valores de K_I. As respostas podem ser visualizadas da Fig. 64 à Fig. 67.

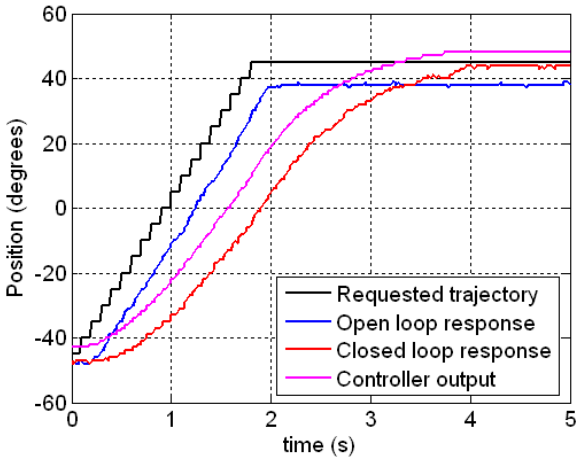

Fig. 64: K_I=0.02 (m=924g)

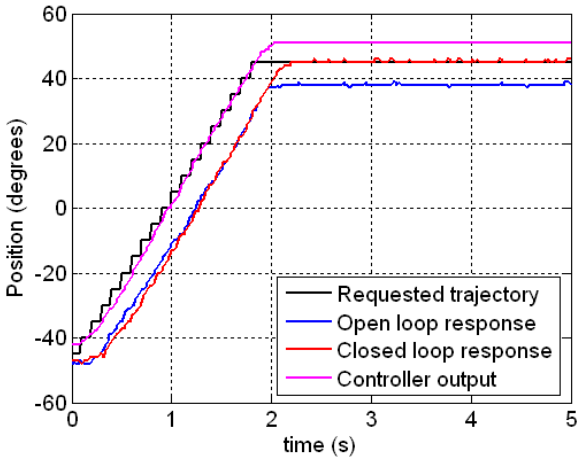

Fig. 65: K_I=0.07 (m=924g)

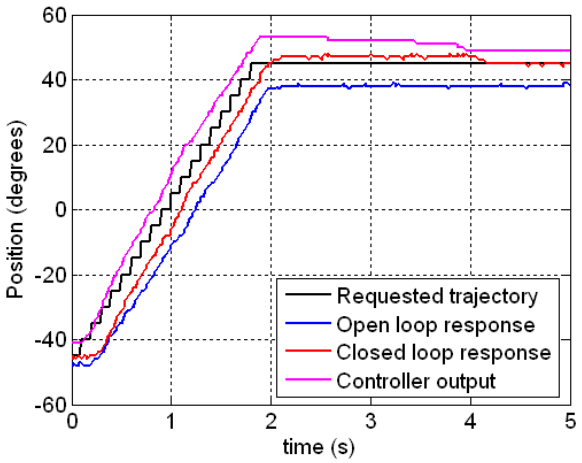

Fig. 66: K_I=0.15 (m=924g)

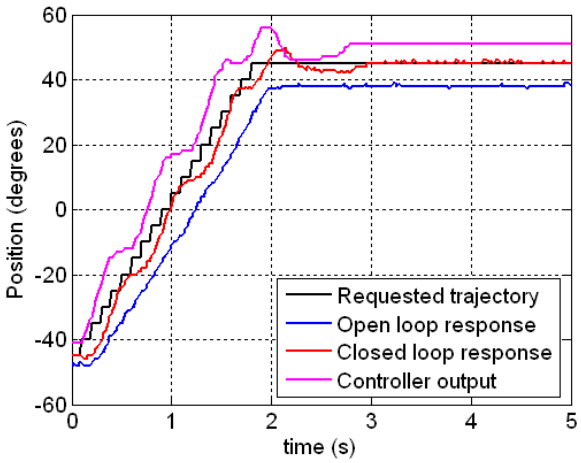

Fig. 67: K_I=0.30 (m=924g)

Por observação, constata-se que para baixos valores, a resposta tende a atrasar-se demasiado, mas à medida

que vai aumentando o tempo de atraso diminui com melhorias significativas relativamente à resposta em malha aberta. O caso da Fig. 67 evidencia a resposta típica para valores excessivos do K_I: durante a fase transitória, tanto a posição medida como o sinal de controlo apresenta-se bastante oscilatório provocando no fim *overshoot*. No entanto seria desejável conter este *overshoot* de modo a melhorar ainda mais o atraso de cerca de 150ms medido pouco antes de se verificar as oscilações.

Quanto ao erro em regime estacionário, em todos os casos apresentados ele é eliminado, o que sugere que é suficiente a presença do integrador, independentemente do parâmetro K_I, exceptuando-se, obviamente, o valor nulo.

3.4.3. Controlo Proporcional+Integral (PI)

Vamos agora adicionar a componente proporcional, mantendo a integral pois é fundamental para a eliminação do erro em regime estacionário.
Para melhor percebermos as vantagens da componente proporcional, primeiramente definamos um K_I de modo a provocar um ligeiro overshoot na resposta do servo. A Fig. 68 e Fig. 69 apresentam um exemplo para uma carga de 924g com overshoot para K_I=0.10 e K_P =0.04.

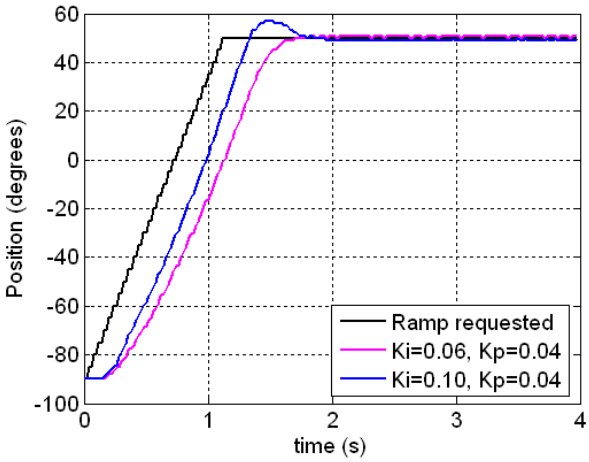

Fig. 68: Fenómeno do overshoot para valores de K_I elevados (carga de 924g)

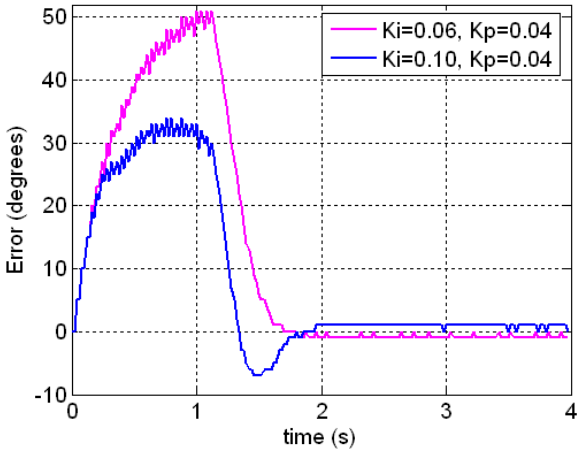

Fig. 69: Erro da resposta da Fig. 55.

Mantendo fixo o valor de K_I em 0.10 vamos aumentar o parâmetro K_P para 0.30 (Fig. 70 e Fig. 71).

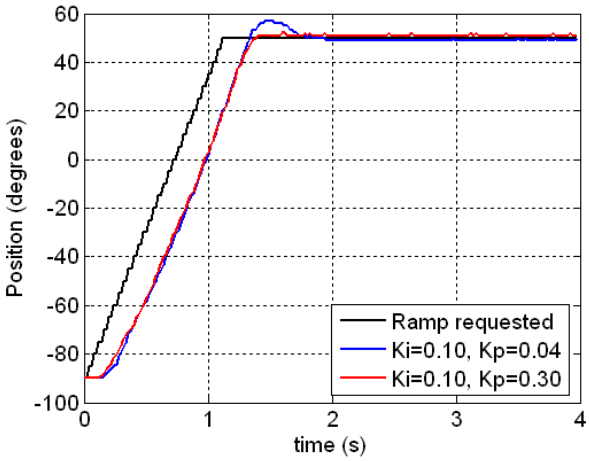

Fig. 70: Correcção do overshoot com o aumento de K_P (carga de 924g).

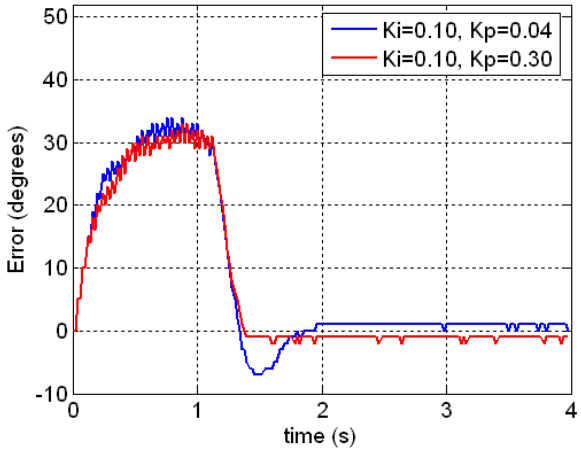

Fig. 71: Erro da resposta da Fig. 57.

Pode-se constatar que o *overshoot* desapareceu sem afectar significativamente o tempo de atraso. De modo a

melhor perceber estes resultados registaram-se várias respostas para diferentes valores de K_P mantendo fixas a trajectória solicitada e a carga. A Fig. 72 e a Fig. 73 apresentam os resultados para dois valores diferentes de K_I agora utilizando trajectórias polinomiais.

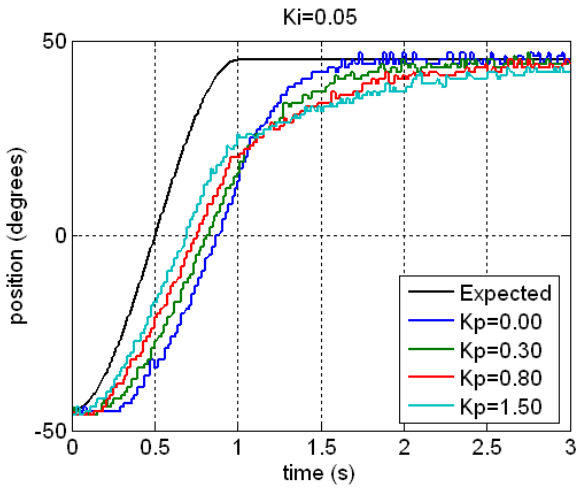

Fig. 72: Variação de K_P para K_I=0.05 (carga de 675g).

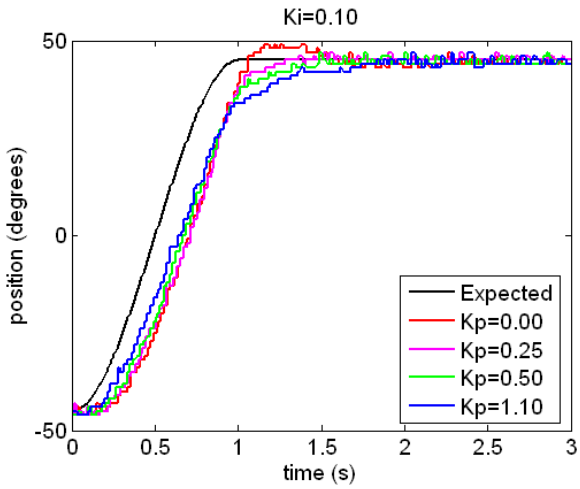

Fig. 73: Variação de K_P para K_I=0.10 (carga de 675g).

Analisando o gráfico da Fig. 73 com K_P=0.10 confirma-se o que se havia dito sobre os efeitos no overshoot: o overshoot é reduzido chegando mesmo a deteriorar o tempo de estabelecimento caso este valor seja excessivo, sem no entanto alterar significativamente o tempo de atraso.
Para um K_I mais pequeno – 0.05 – (Fig. 72), pode-se evidenciar mais claramente os efeitos de um K_P excessivo em valor: o tempo de estabelecimento é claramente prejudicado levando muito mais tempo a atingir o valor final, mas o tempo de atraso é melhorado acelerando o seguimento da trajectória no seu início. Ocorre, por isso, um ponto de cruzamento entre as diversas respostas quase coincidente com o término da trajectória.
Logo, conclui-se que deve haver um compromisso entre o tempo de atraso e o tempo de estabelecimento de modo a não perder demasiado num dos lados. O parâmetro K_P deve, por isso, ser escolhido tendo em conta estes aspectos.

3.4.4. Controlo Integral+Derivativo (ID)

Substituindo a componente proporcional pela derivativa podemos observar que praticamente não afecta a acção integral, tal como se observar na Fig. 74 e na Fig. 75, pelo que não se encontra utilidade para este tipo de controlo.

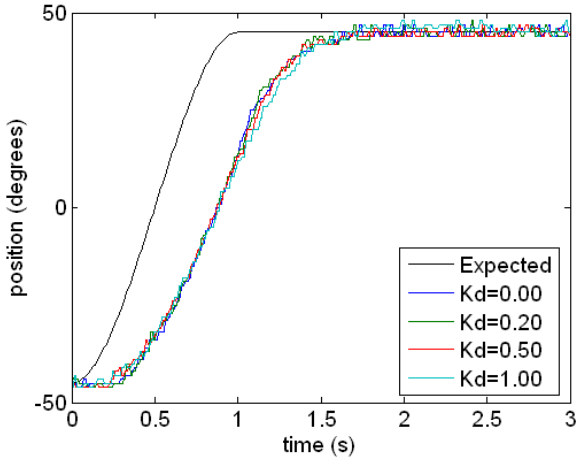

Fig. 74: Resposta ao polinómio com K_I=0.05 e K_P=0.00 (carga de 675g).

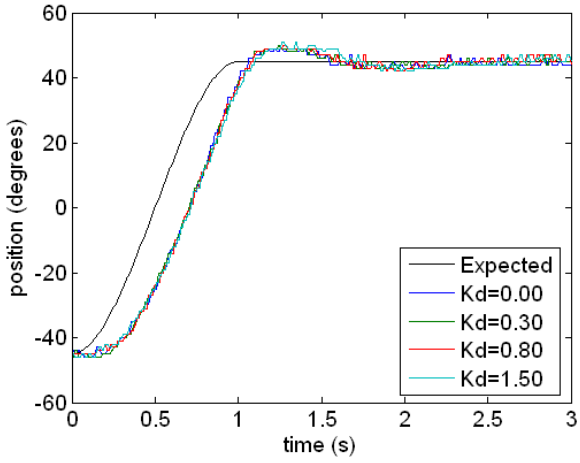

Fig. 75: Resposta ao polinómio com K_I=0.10 e K_P=0.00 (carga de 675g).

3.4.5. Controlo Proporcional+Integral+Derivativo (PID)

Introduzindo agora todas as três componentes podemos optimizar todos os parâmetros de modo a encontrar a melhor relação entre:

- ➔ Tempo de atraso;
- ➔ Tempo de estabelecimento;
- ➔ Overshoot;
- ➔ Oscilação durante a fase transitória.

Por análise de cada componente do controlador, apresentadas nas secções anteriores, já podemos relacionar estes parâmetros com cada um deles:

- ➔ A **componente integral** está directamente relacionada com o tempo de atraso, sendo tanto menor quanto maior for o parâmetro K_I. No entanto para valores excessivos começa a instabilizar com o surgimento de *overshoot*.
  Resumindo:
  - Diminui o tempo de atraso;
  - Aumenta o *overshoot* para valores de K_I elevados.
- ➔ A **componente proporcional** é utilizada para contenção do *overshoot*, sendo capaz também de melhorar o tempo de atraso se K_P possuir valores elevados, sob pena do tempo de estabelecimento se deteriorar.
  Logo, temos:
  - Redução do *overshoot*;
  - Para valores elevados de K_P:
    - O tempo de atraso é melhorado;
    - O tempo de estabelecimento deteriora-se.
- ➔ Embora ainda não tenha sido demonstrado a utilidade da **componente derivativa**, acredita-se que ela é capaz de conferir estabilidade durante a fase transitória, pelo que, para valores limite da compensação PI será interessante incluir a componente derivativa para acréscimo da estabilidade.

Combinando estas três componentes deveremos conseguir uma resposta muito próxima da ideal, com um tempo de atraso muito pequeno, um tempo de estabelecimento próxima da duração da trajectória, overshoot nulo e um comportamento suave durante a fase transitória.

Como procedimento para fazer o tunning da compensação seguiram-se os passos seguintes:

1. Aumentar K_I, de modo a optimizar o tempo de atraso, até começar a ocorrer *overshoot*;
2. Aumentar o valor de K_P o suficiente para eliminar o *overshoot*. Não convém utilizar este parâmetros para optimizar o tempo de atraso, uma vez que o tempo de estabelecimento é, ao mesmo tempo, agravado. Deixemos, por isso, essa tarefa à acção integral;
3. Se a resposta transitória ainda não for demasiado oscilante, voltar ao passo 1 para melhorar ainda mais o tempo de atraso;
4. Se começar a instabilizar durante a fase transitória, aumentar o parâmetro K_D de modo a conferir suavidade durante o percurso;
5. Voltar ao passo 1.

A Fig. 76 compara dois ensaios executados durante o tunning correspondentes a um ajuste inicial e a outro final, verificando-se uma melhoria de cerca de 6º no erro máximo em regime transitório. A Fig. 77 apresenta um caso de exagero nos parâmetros de compensação levando à instabilidade. De modo a evitar estas situações convém executar o algoritmo de *tunning* em passos pequenos, permitindo assim encontrar os parâmetros óptimos mais facilmente.

A partir de certo ponto, ao qual chamaremos de compensação limite, a melhoria já começa a ser mais exigente com variações muito mais pequenas dos parâmetros de compensação. Quando tal começa a ocorrer considere o *tunning* como terminado com os parâmetros óptimos correspondentes ao ajuste anterior.

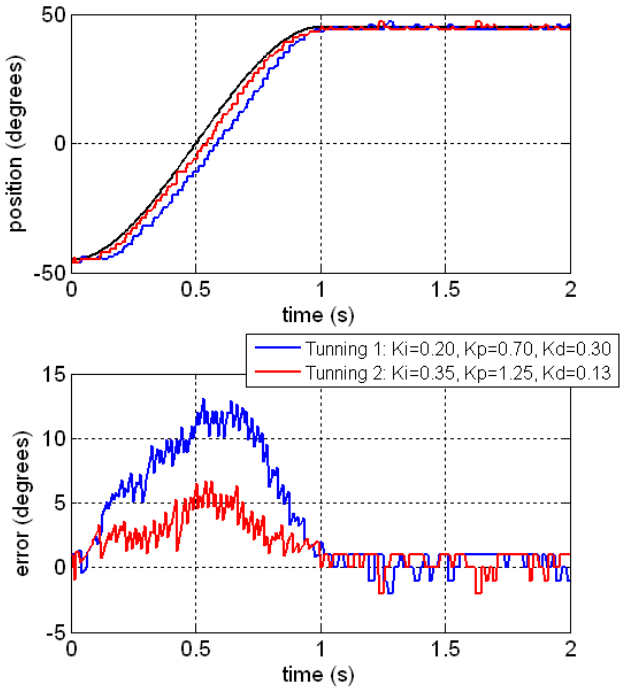

Fig. 76: Comparação entre dois conjuntos de parâmetros durante o tunning (m=675g).

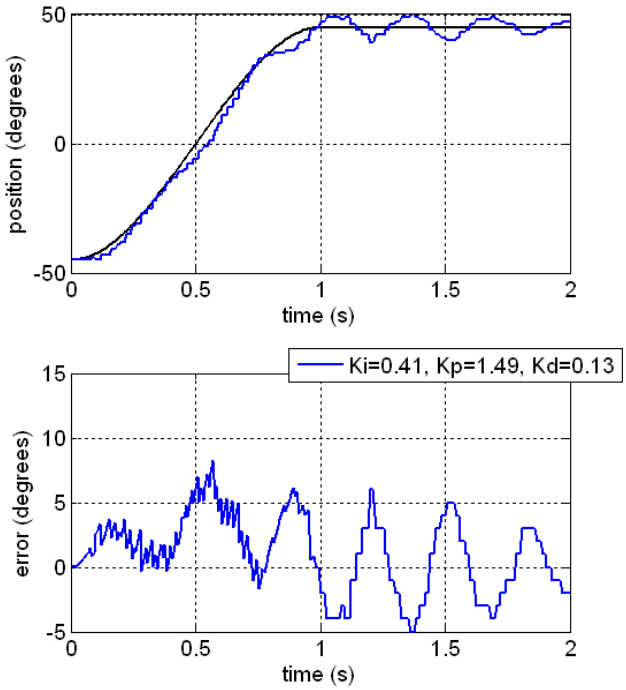

Fig. 77: Situação de instabilidade (m=675g).

Para a carga de 675g encontraram-se como parâmetros óptimos o conjunto K_I=0.39, K_P=1.46 e K_D=0.15 com um erro máximo de menos de 5º (Fig. 78). No entanto, note que nos encontramos numa situação em que o sistema torna-se muito susceptível à instabilidade face a perturbações externas. A Fig. 79 exemplifica este caso, em que o ensaio com os parâmetros óptimos é repetido verificando-se agora alguma instabilidade.

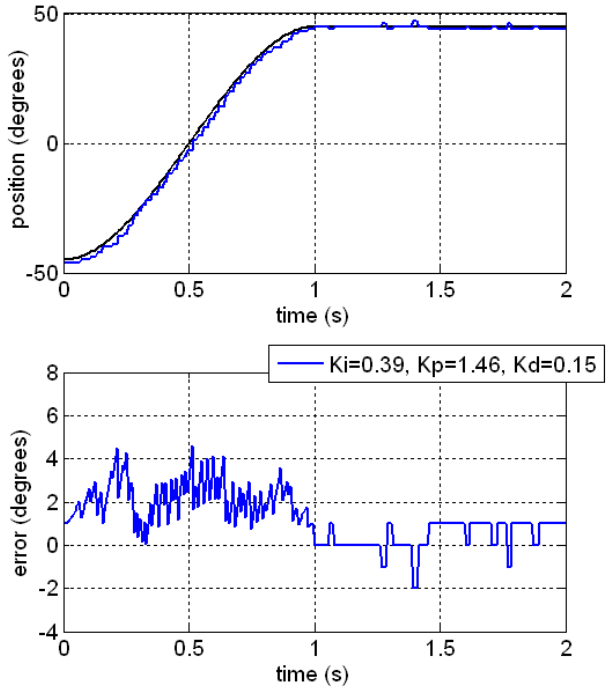

Fig. 78: PID optimizado para uma carga de 675g.

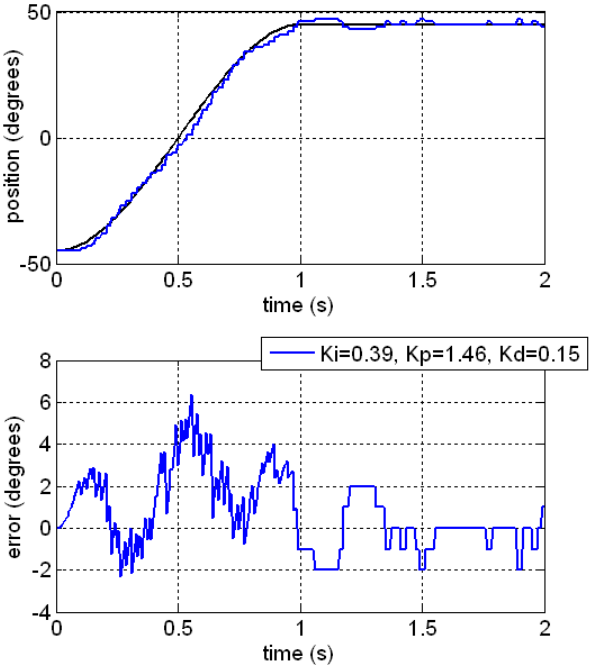

Fig. 79: Repetição do ensaio com os parâmetros de compensação óptimos.

3.4.6. Algumas Notas

Embora a solução de controlo baseada num PID aparente ter bons resultados, não dos devemos esquecer que determinado um conjunto óptimo de parâmetros para compensação dos desvios do servo, apenas se aplica nas condições em que foi feito o *tunning*:

- ➔ Massa da carga;
- ➔ Posição inicial e final, ou melhor dizendo, variação do binário ao longo da travessia;
- ➔ Período do trajecto;
- ➔ Tensão de alimentação e corrente máxima fornecida.

Mudando qualquer um destes parâmetros resulta invariavelmente na alteração da resposta do servo podendo conferir-lhe maior estabilidade ou torná-lo instável:

- ➔ O aumento da massa da carga induz à instabilidade, uma vez que o servo tem de fazer um "esforço" maior para a mover, no sentido que tem de aplicar maior binário no motor, e por isso, o tempo de atraso será acrescido. Numa situação destas em que o sinal de *feedback* tem dificuldade em acompanhar a "velocidade" do controlador (definido pelo factor de integração K_I), o sinal de controlo tende a saturar facilmente provocando fenómenos de *overshoot*. Por outro lado, se a carga diminuir de massa, o sinal de *feedback* consegue acompanhar perfeitamente o sinal de controlo conferindo uma maior reactividade à sua correcção.
- ➔ Se o percurso da trajectória for alterado, a trajectória de binário também muda fazendo com que o servo tenha de dispender mais ou menos energia, de acordo com o caso, para a realizar. Esta situação é equivalente à da modificação da carga, na medida que quando o binário a que está sujeito aumenta (aproxima-se do ponto 0º) é equivalente a aumentar a massa da carga, e vice-versa.
- ➔ Além da carga influenciar a resposta do servo, também a velocidade influencia. Tal é devido ao facto, de não só a força gravítica fazer parte do binário exercido no servo, mas também a velocidade e a sua variação (aceleração). Desta forma, diminuindo o período da trajectória, estamos a aumentar a velocidade o que interfere no binário exercido no induzindo uma resposta mais instável.
- ➔ Um outro detalhe importante são as condições de alimentação eléctrica dos servomotores. Caso a corrente máxima a fornecer seja limitada ou a tensão de alimentação é baixa, o binário a aplicar no motor para executar um determinado movimento aumenta, introduzindo atraso na resposta e logo maior instabilidade. Para minimizar estes problemas, duas baterias de Lítio de 7.4V são ligadas em paralelo de modo a fornecer uma corrente de 9600 mAh ao sistema.

Como se pode constatar, estamos a lidar com um processo altamente não linear em que as condições iniciais aplicadas nos servomotores estão sempre a mudar, o que é um problema, uma vez que o controlo clássico baseado num PID não entra em consideração com as condições iniciais.

Uma forma de dar a volta a esta questão corresponde a actualizar em tempo real o valor dos parâmetros de compensação de modo a adaptar o controlador a cada situação específica. Foi pensando nesta questão que se decidiu que a compensação via *software* seria a melhor opção, uma vez que é muito fácil mudar os parâmetros de controlo sem intervenções a nível de hardware como acontecia se o controlador estivesse implementado fisicamente. De modo a evitar a modificação do código na modificação destes valores, os parâmetros de controlo são passados a cada *slave* via barramento CAN, sendo a unidade principal, o PC, a responsável por atribuir os valores de compensação mais apropriados a cada acção.

No entanto, outro problema surge: como é que se detectam as situações mais ou menos exigentes em cada junta; e caso consigamos detectá-las, que lei de controlo seguirão os parâmetros de compensação? Para já tentaremos responder à primeira questão. Muito embora, no teste de um só servo, haja uma relação estreita entre binário aplicado e posição, tal deixa de acontecer na presença de várias juntas que se interligam em série por meio de elos, como é o que acontece com cada perna. Além disso, a velocidade da junta também afecta o binário pelo que é preciso discernir cada uma destas fontes de binário. Uma das formas de estimação do binário aplicado baseia-se na medição da corrente consumida por cada servomotor: quanto maior for o binário aplicado, maior é a corrente consumida pelo que a medição desta grandeza pode ajudar na detecção de situações de elevado ou de baixo *stress* sobre a juntas.

3.5. MONITORIZAÇÃO DE CORRENTE

3.5.1. Estudo Estático da Corrente

Comecemos por avaliar a corrente consumida em situações de carácter estático, ou seja, com o servomotor em repouso, analisando-a de acordo com o binário aplicado. A Fig. 80 e a Fig. 81 apresentam um conjunto de medições de corrente realizadas com o servo em repouso para cada uma das posições avaliadas.

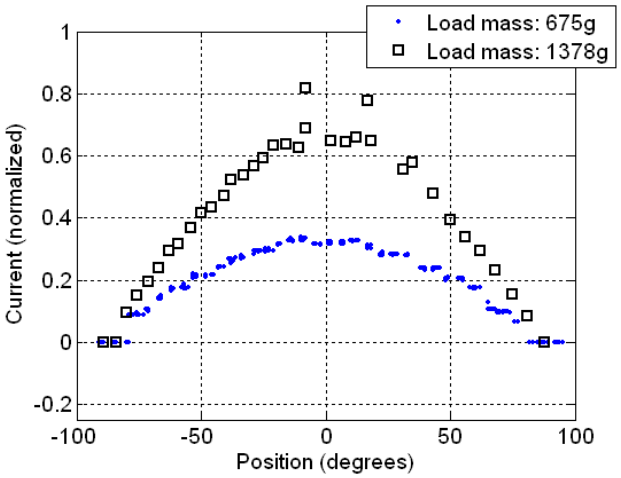

Fig. 80: Medição estática da corrente para duas cargas num percurso em subida (-90º a +90º).

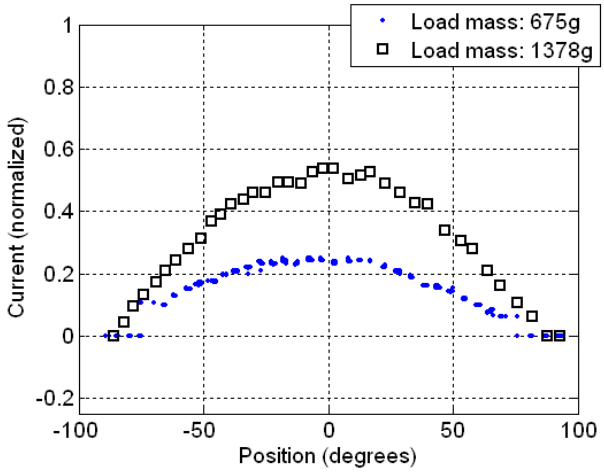

Fig. 81: Medição estática da corrente para duas cargas num percurso em descida (+90º a -90º).

Como não entram nos resultados o factor velocidade, estes apenas dependerão da força gravítica o que será de esperar que o valor máximo de corrente corresponda à posição 0º (máximo binário) e os valores mínimos aos extremos ±90º (binário nulo), tal como nos mostram os gráficos. Comparando com duas massas diferentes verifica-se que quanto maior é a massa da carga, maior é a corrente consumida evidenciando a estreita relação da força gravítica com a corrente consumida. No entanto note a diferença numérica dos resultados quando a medição se processo no sentido descendente (+90 para -90º) comparativamente ao oposto: na descida a corrente consumida é menor, talvez devido ao facto de os movimentos exigirem menor esforço que no outro caso... no entanto tal ainda não está completamente esclarecido uma vez que estamos a estudar o servo em repouso e não em movimento.

3.5.2. Estudo Dinâmico em Malha Aberta

Introduzindo agora o elemento velocidade, fizemos vários capturas do consumo de corrente com o servo em pleno movimento – estudo dinâmico da corrente. Começámos por realizar movimentos seguindo a trajectória polinomial sem o controlador PID, ou seja, em malha aberta.
Por questões de rigor não vamos caracterizar cada trajectória a partir das posições inicial e final, uma vez que o que nos interessa é a variação de binário resultante da força gravítica. Aliás, as posições deixam de ter significado quando a configuração do servo é modificada para além da indicada na Fig. 32, pelo que referiremos o binário gravítico inicial, final e intermédio com a equivalência do percurso usando posições de referência segundo a configuração original:

- ➔ Trajectória desde o ponto de binário nulo até ao máximo: equivalente ao trajecto de ±90º para 0º em subida ou em descida;
- ➔ Trajectória desde o ponto de máximo binário até ao nulo: equivalente ao trajecto de 0º até ±90º em subida ou em descida;
- ➔ Trajectória entre pontos de binário intermédio passando pelo valor máximo: equivalente ao trajecto entre -45º e +45º.

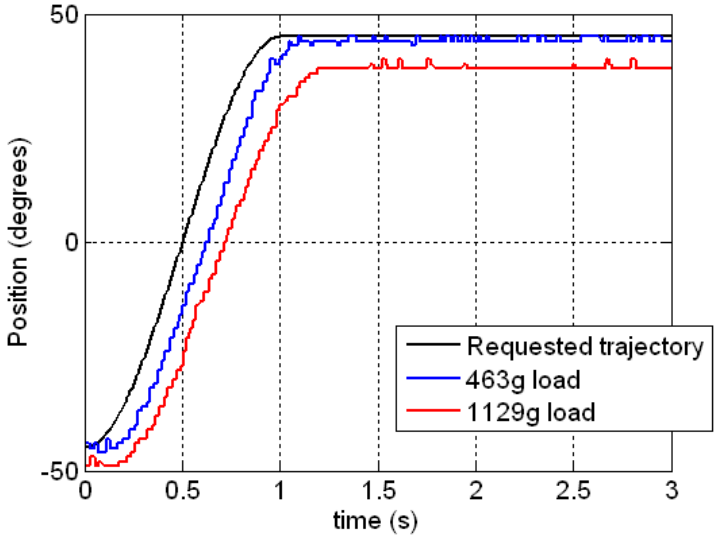

Fig. 82: Resposta em malha aberta de uma trajectória polinomial de 1s entre dois pontos de binário intermédio para duas cargas.

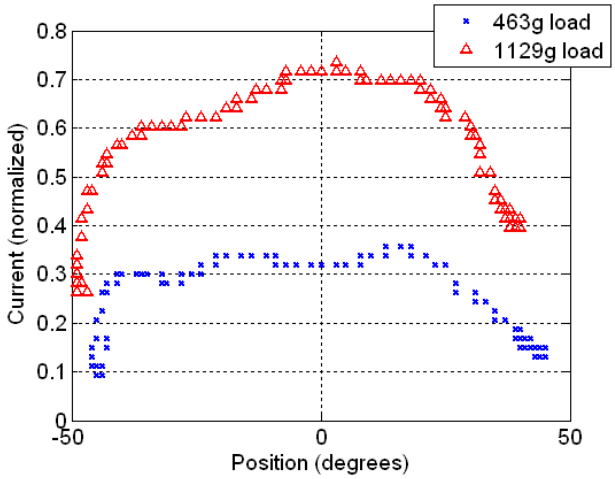

Fig. 83: Medição da corrente para cada posição do trajecto da Fig. 82.

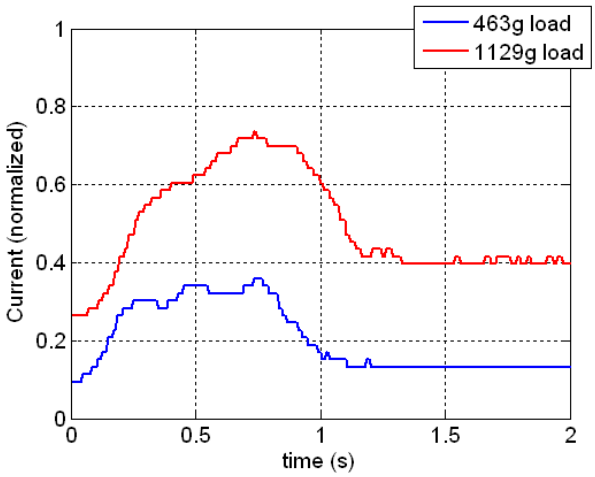

Fig. 84: Medição da corrente ao longo do tempo do trajecto da Fig. 82.

Realizando movimentos entre dois pontos de binário intermédio passando pelo valor máximo (equivalente ao percurso desde -45º a +45º na configuração original do servo) continua-se a verificar um comportamento corrente *versus* posição (Fig. 83) muito semelhante ao caso estático (Fig. 80), mas com a adicionante de ocorrer uma maior oscilação na distribuição ao longo das posições. A Fig. 84 mostra-nos o comportamento ao longo do tempo evidenciado este aspecto.

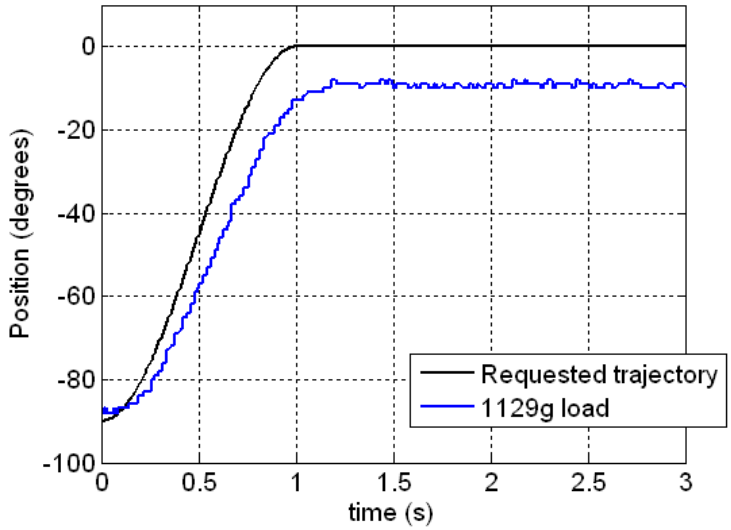

Fig. 85: Resposta em malha aberta de uma trajectória polinomial de 1s desde o ponto de binário nulo até ao máximo.

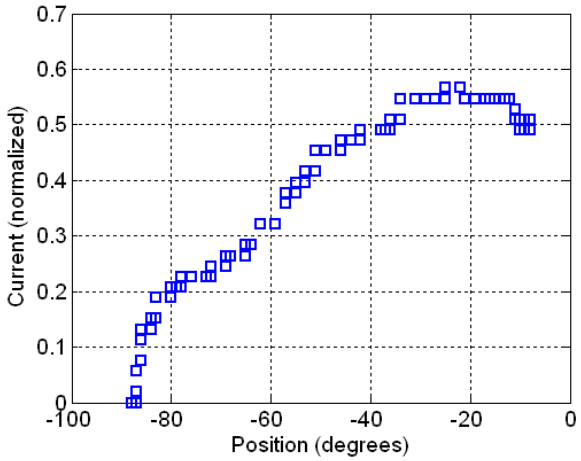

Fig. 86: Medição da corrente para cada posição do trajecto da Fig. 85.

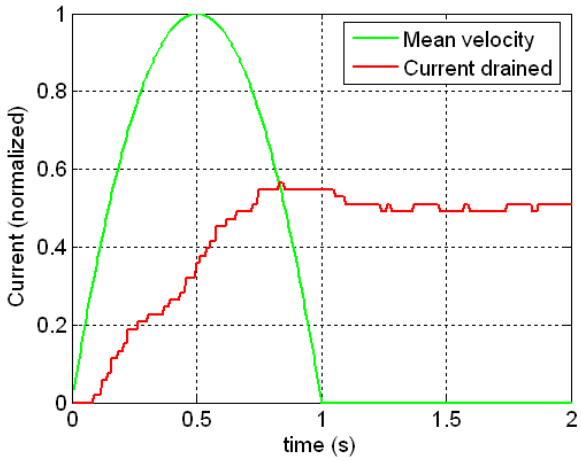

Fig. 87: Medição da corrente ao longo do tempo do trajecto da Fig. 85.

Aplicando um trajecto desde um ponto de binário nulo até ao máximo (-90º até 0º), podemos verificar na Fig. 86 e na Fig. 87 o crescente consumo de corrente à medida que o binário aumenta. No entanto, note numa ligeira descida da corrente quando a trajectória termina: como a força gravítica não diminui neste ponto, a causa provável pode estar residente na velocidade do servo. Até ao fim do percurso temos a contribuição da força gravítica e da velocidade no binário aplicado ao servo, mas no fim apenas temos a força gravítica o que pode justificar o decréscimo da corrente.

De modo a estudarmos melhor este fenómeno, realizámos uma segunda experiência, desta vez com uma trajectória iniciando num ponto de máximo binário, e terminando em binário nulo (equivalente ao percurso desde 0º até 90º na configuração original) (Fig. 88).
Seria de esperar que o início da resposta correspondesse à máxima corrente, uma vez que é neste ponto em que a componente gravítica do binário é mais significativa, e fosse descendo até zero quando chegasse ao ponto de binário nulo. No entanto, pela Fig. 90, não é isso o que se verifica: embora no instante inicial já esteja em consumo uma corrente não pouco significativa, após o arranque do movimento, ela aumenta ainda mais atingindo um máximo no ponto coincidente à máxima velocidade. Só a partir daqui é que a corrente começa a baixar continuamente até zero.

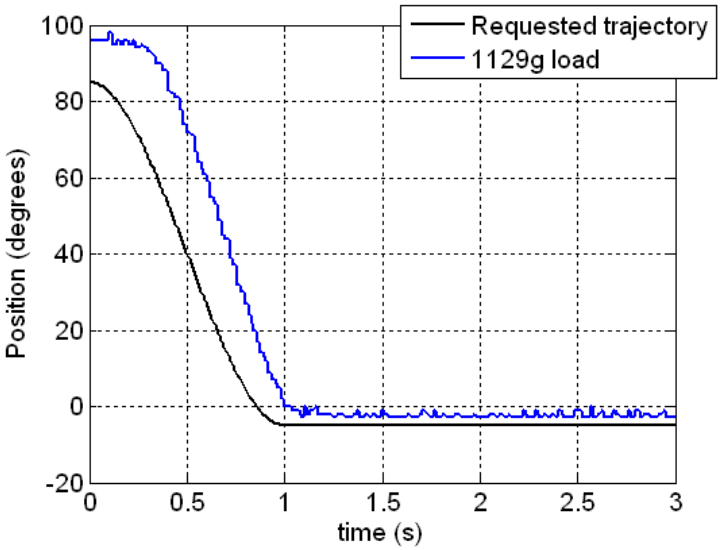

Fig. 88: Resposta em malha aberta da trajectória polinomial de 1s desde o ponto de máximo binário até ao nulo.

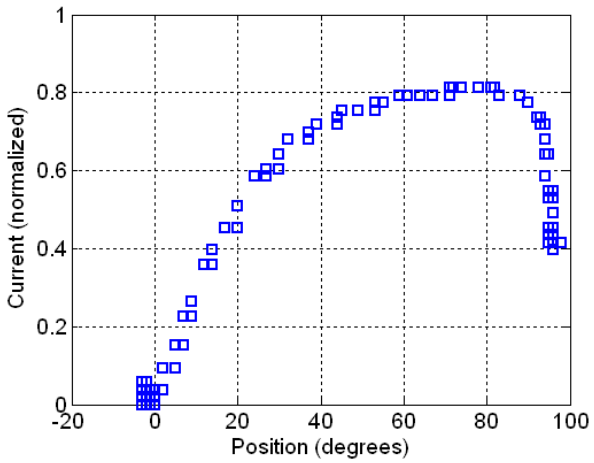

Fig. 89: Corrente consumida em cada posição do trajecto da Fig. 88.

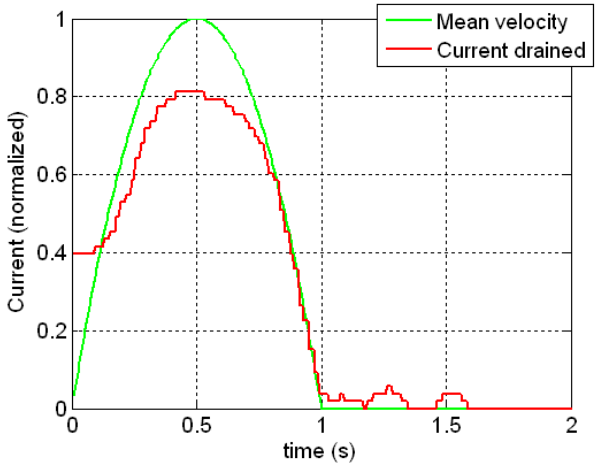

Fig. 90: Corrente consumida ao longo do tempo do trajecto da Fig. 88.

Este comportamento indica-nos claramente que a força gravítica não é a única que contribui na corrente consumida, contribuindo também a velocidade e a aceleração.

$\tau_{motor} = \tau_{gravítico} + \tau_{velocidade} + \tau_{aceleração}$ com $\tau_{motor} = K \cdot I$

Uma questão surge, entretanto. Apenas conseguimos medir o binário total aplicado ao motor, mas, no entanto, apenas nos interessa o resultante da força gravítica, pois é a componente que nos indica a inércia a que está submetida. Para conseguirmos isolar esta informação precisaríamos de conhecer a velocidade e aceleração em cada instante, uma vez que se relaciona com o respectiva componente de binário de forma proporcional:

$$\tau_{gravítico} = m \cdot g \cdot L \cdot \sin(\theta)$$
$$\tau_{velocidade} = f_m \cdot \dot{\theta}, \quad f_m \rightarrow atrito\ do\ motor$$
$$\tau_{aceleração} = J_m \cdot \ddot{\theta}, \quad J_m \rightarrow Inércia\ do\ motor$$

No entanto, apenas conseguimos fazer uma estimativa da velocidade em cada 100ms (muito sujeita a erro) e não podemos medir a aceleração, além que desconhecemos os parâmetros K, f_m e J_m para podermos relacionar as diversas componentes. Por enquanto este problema ainda carece de solução.

3.5.3. Estudo Dinâmico em Malha Fechada

Nas análises seguintes vamos considerar agora o controlador PID ligado, ensaiando os mesmos trajectos que foram aplicado em malha aberta. Vemos então verificar se o consumo de corrente sofre variações significativas devido ao controlo.

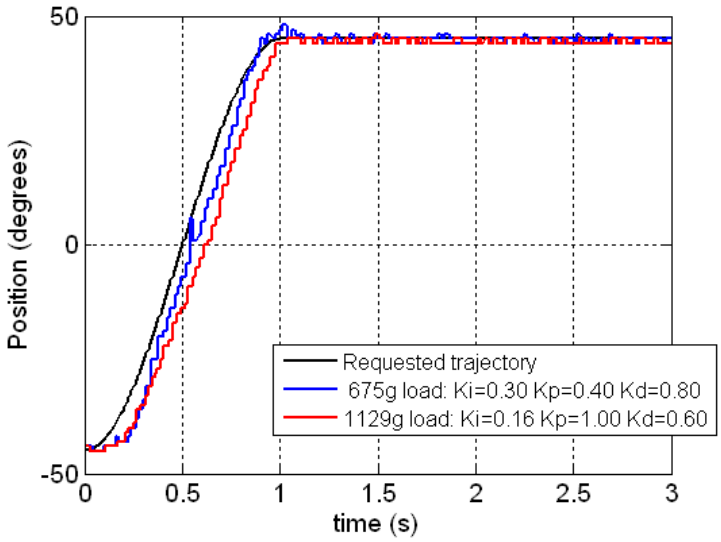

Fig. 91: Resposta em malha fechada de uma trajectória polinomial de 1s entre dois pontos de binário intermédio.

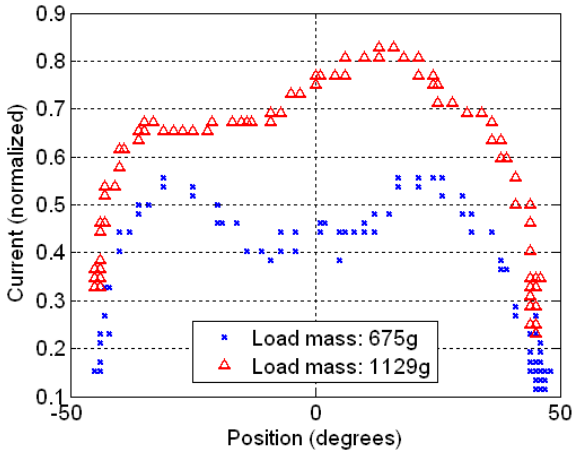

Fig. 92: Corrente consumida em cada posição do trajecto da Fig. 91.

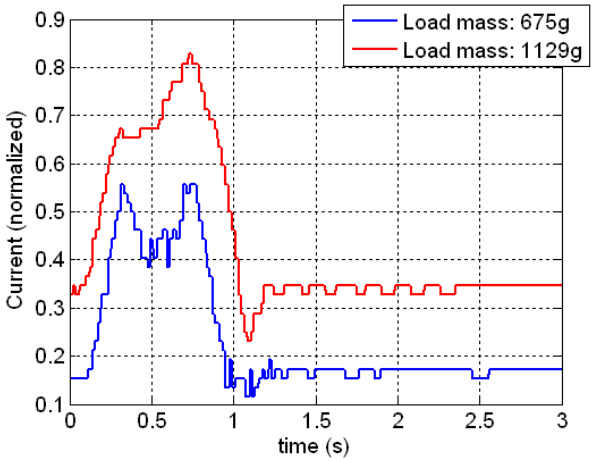

Fig. 93: Corrente consumida ao longo do tempo do trajecto da Fig. 91.

Optimizando o controlador para as condições iniciais:

- Trajectória: entre pontos de binário intermédio (-45 a +45º);
- Velocidade correspondente ao período de 1s;
- Cargas de 675g e de 1129g.

Medimos a resposta em termos de corrente, e pode-se verificar pela Fig. 92 e pela Fig. 93 uma oscilação ainda mais notória da corrente consumida. Tal é facilmente explicável pelas variações do sinal de controlo para compensar a posição solicitada: agora já não se trata de uma variação polinomial como acontecia em malha aberta, mas dependente do sinal de erro entre o sinal de *feedback* e a trajectória polinomial desejada, o que pode resultar num consumo de corrente mais irregular. A resposta ao longo do tempo evidencia claramente este consumo irregular.

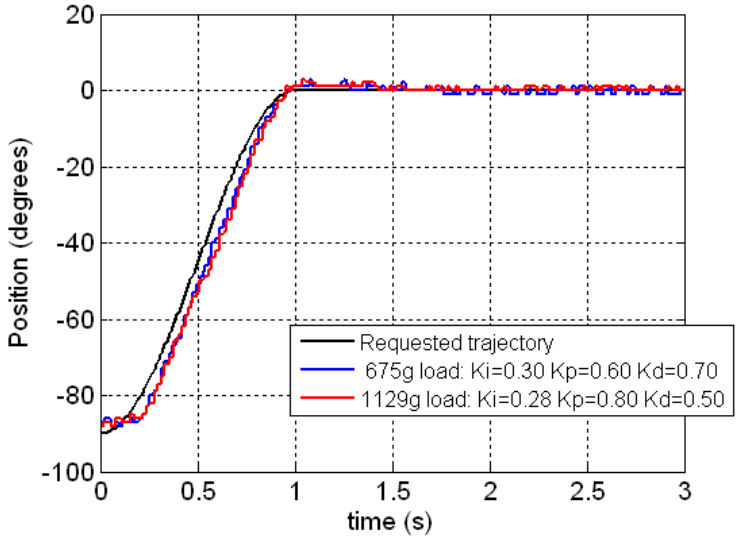

Fig. 94: Resposta em malha fechada de uma trajectória polinomial de 1s desde o ponto de binário nulo até ao máximo.

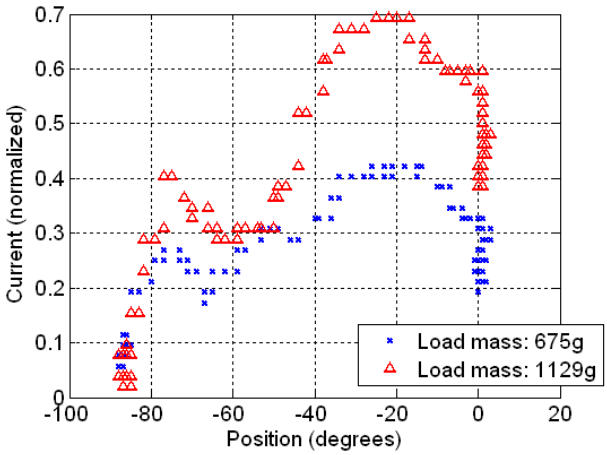

Fig. 95: Corrente consumida em cada posição do trajecto da Fig. 94.

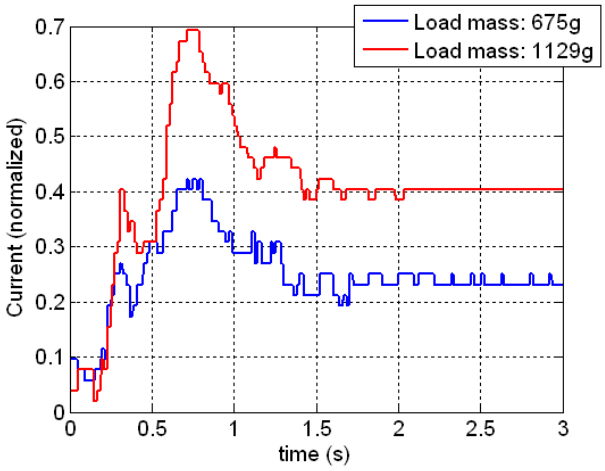

Fig. 96: Corrente consumida ao longo do tempo do trajecto da Fig. 94.

Para a trajectória do mínimo para o máximo binário (-90º a 0º) o consumo de corrente ainda é mais irregular, talvez devido ao crescente binário aplicado em toda o trajecto, fazendo com que o servo tenha mais dificuldades em acompanhar a taxa de integração do controlador. Tal resulta numa maior variação do sinal de controlo aumentando, por isso, a instabilidade do consumo de corrente.

Em jeito de conclusão, verificamos que o consumo de corrente com o controlador presente é muito mais oscilante trazendo bastantes problemas na sua interpretação, o que dificulta a procura por uma lei de variação dos parâmetros de controlo. Por enquanto este problema continua sem solução à vista, ainda mais pela dificuldade em extrair o binário resultante da força gravítica.

3.6. APLICAÇÃO DOS ALGORITMOS NO ROBOT HUMANÓIDE

Conectando o microcontrolador, com os algoritmos de controlo implementados, a três servomotores correspondentes às seguintes juntas de uma perna humanóide:

- ➔ Servo 1: junta do pé (de rotação dianteira);
- ➔ Servo 2: junta do joelho;
- ➔ Servo 3: junta da anca (de rotação dianteira).

... executámos vários movimentos, no qual apresentamos neste capítulo apenas o de flexão (Fig. 97), dada a sua relevância na realização de um passo. Inicialmente fizémo-lo sem carga, e posteriormente adicionámos uma carga de cerca de 2Kg ao topo da perna (Fig. 98).

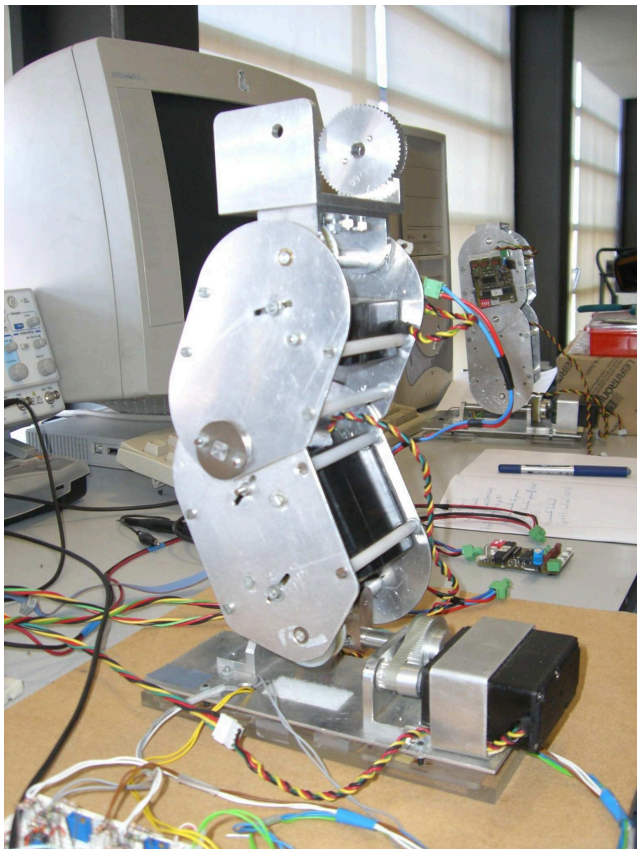

Fig. 97: Movimento de flexão de uma perna.

Fig. 98: Carga de 2Kg utilizada no topo da perna.

3.6.1. Movimento de Flexão em Malha Aberta de uma Perna

3.6.1.1. Na Ausência de Carga:

Como se pode visualizar nas figuras seguintes, na ausência de carga, o erro em regime estacionário é praticamente nulo, praticamente sem a necessidade de praticar controlo externo.

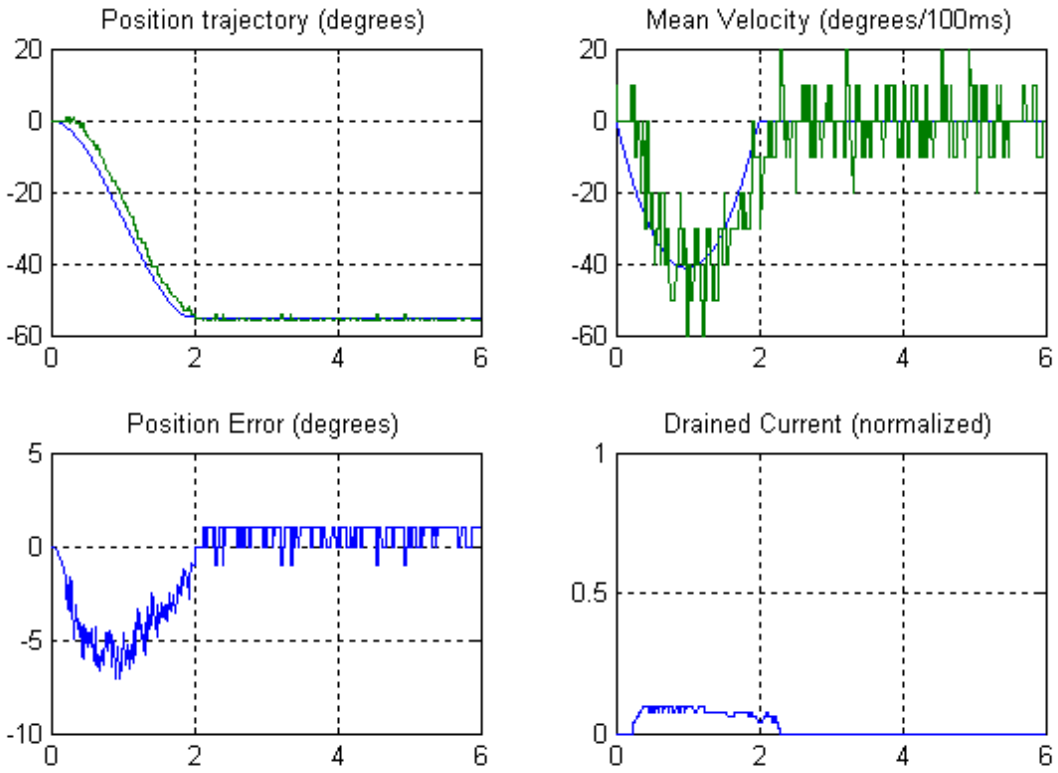

Fig. 99: Junta do pé.

Note na natureza errática na estimação da velocidade. Tal deve-se ao processo de medição baseada na variação de posição, que infelizmente introduz bastante erro.

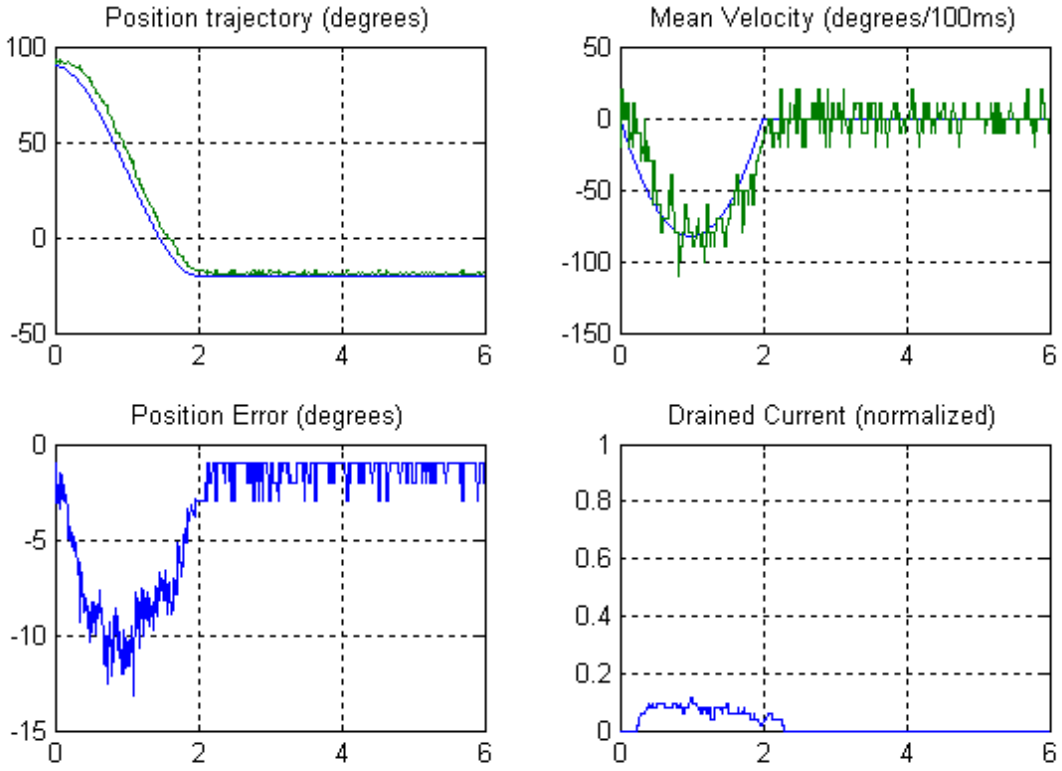

Fig. 100: Junta do Joelho.

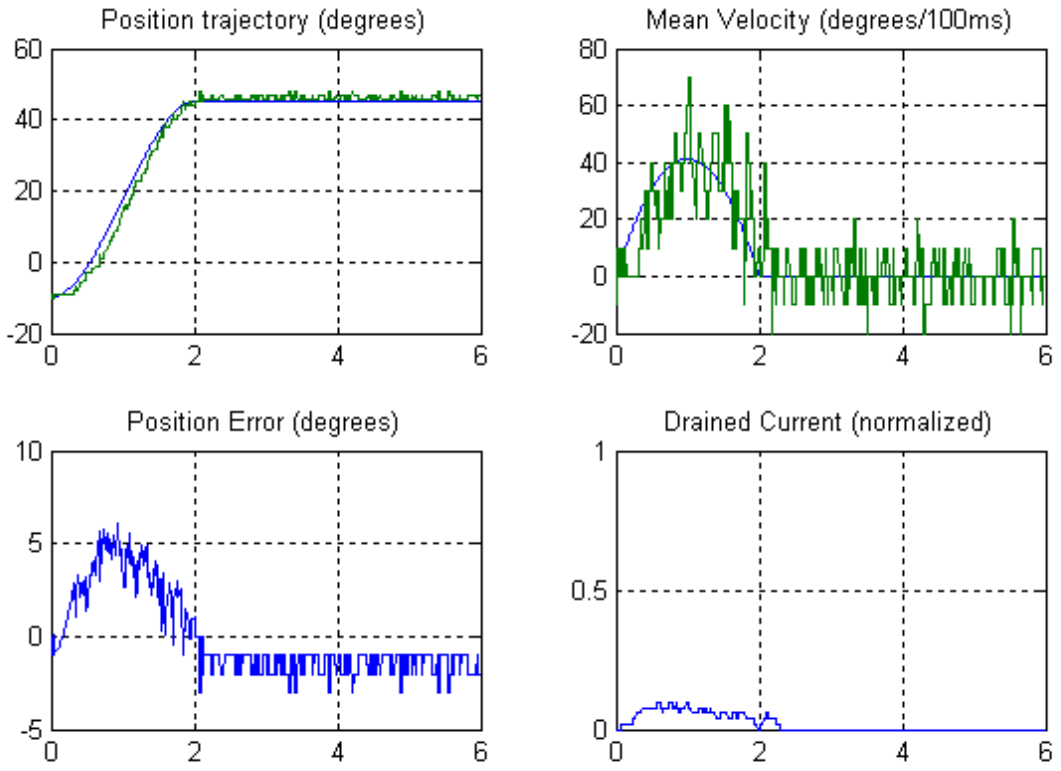

Fig. 101: Junta da Anca.

3.6.1.2. Na presença de uma Carga de cerca de 2Kg:

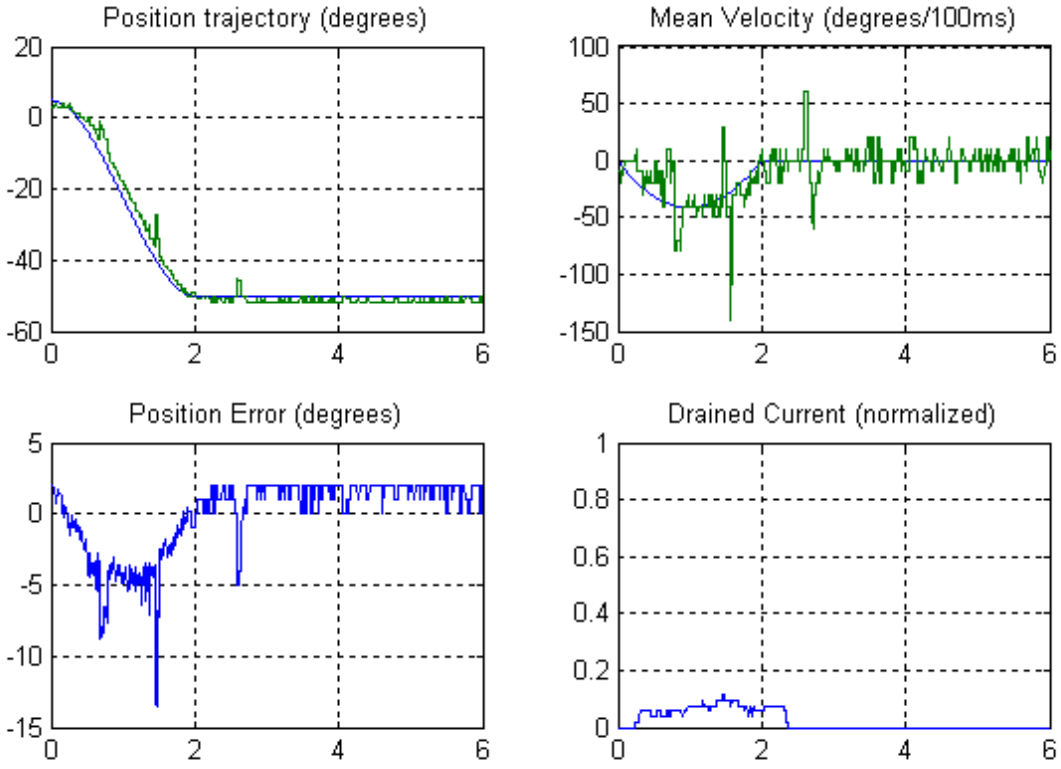

Fig. 102: Junta do pé.

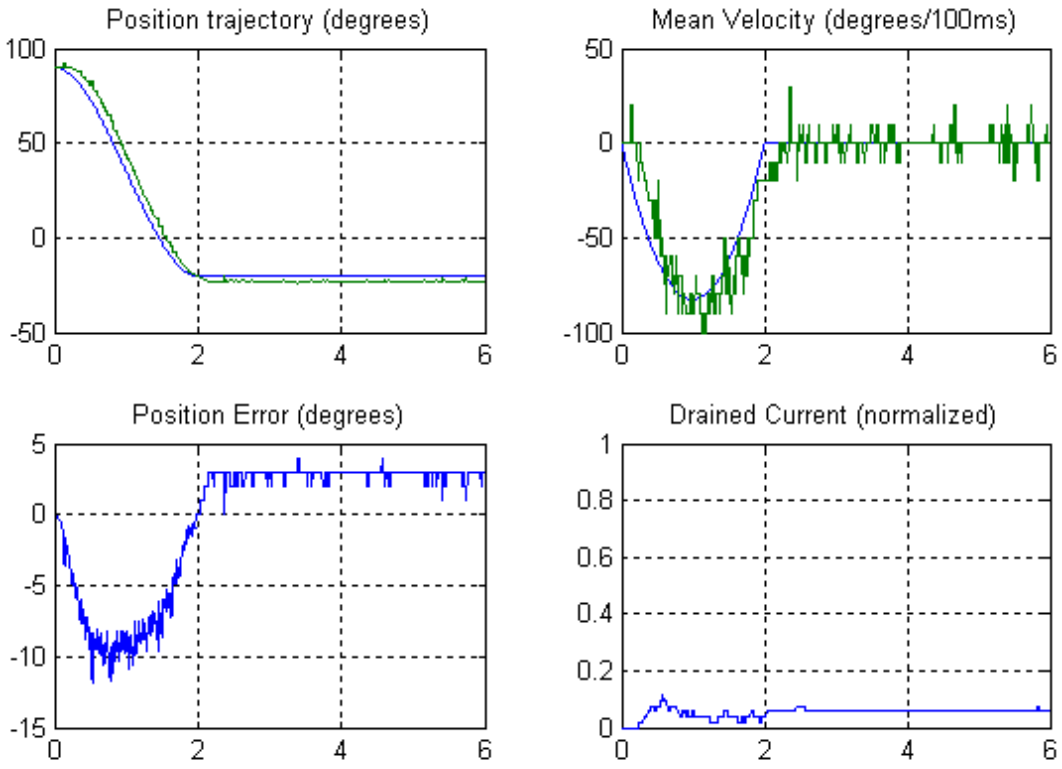

Fig. 103: Junta do joelho.

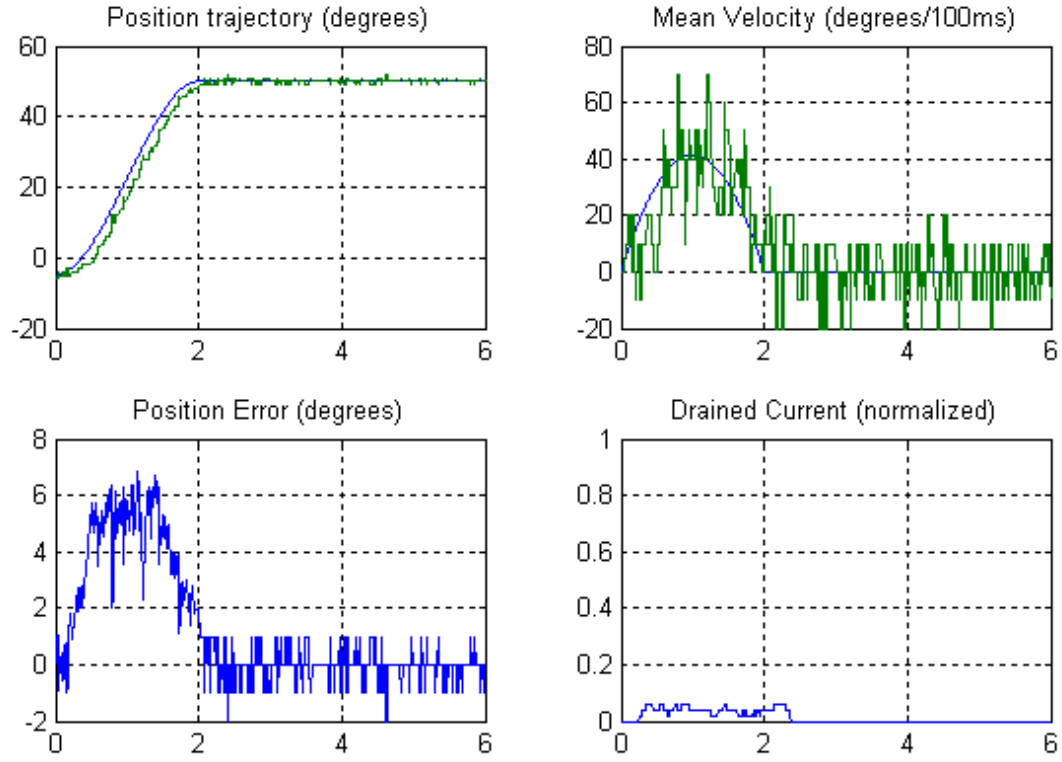

Fig. 104: Junta da anca.

Com a carga de 2Kg, surpreendentemente verifica-se pouca variação do erro em regime estacionário, com um baixo consumo de corrente. Note que agora o esforço é distribuído pelas três juntas, pelo que as condições não são tão exigentes como as utilizadas no capítulo anterior.

3.6.2. Movimento de Flexão em Malha Fechada de uma Perna

As figuras seguintes apresentam o comportamento das juntas, agora com o controlador ligado.
O caso mais exigente verifica-se no caso da junta do pé (Fig. 105), com erro em regime transitório de 14º e em regime estacionário de 5º. Com o controlador ligado, a resposta melhorou bastante reduzindo o erro máximo em regimen transitório para menos de metade, e eliminando o erro em regime estacionário.

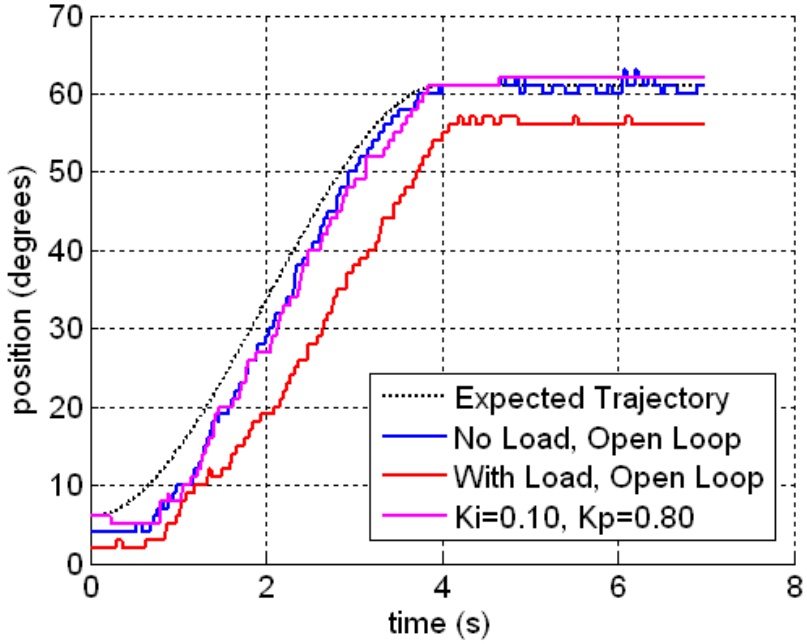

Fig. 105: Junta do pé.

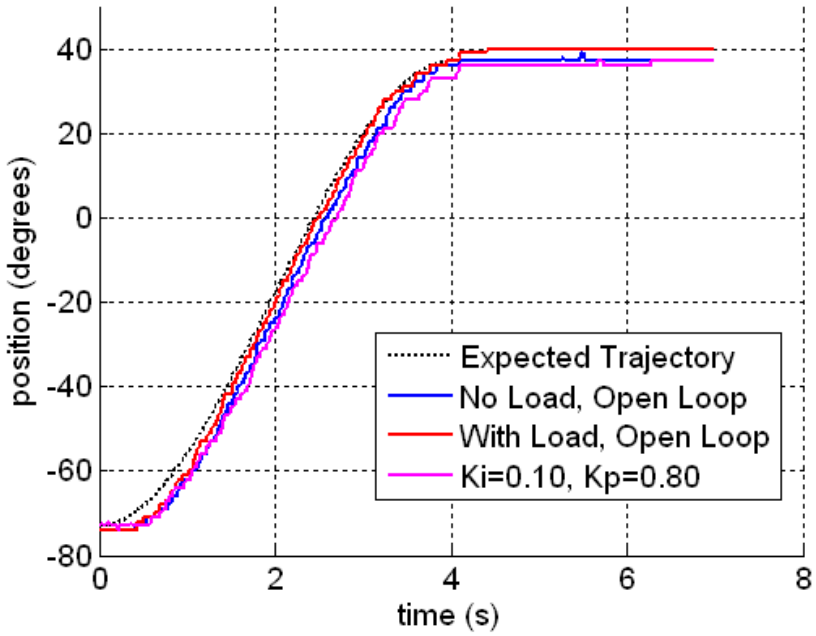

Fig. 106: Junta do joelho.

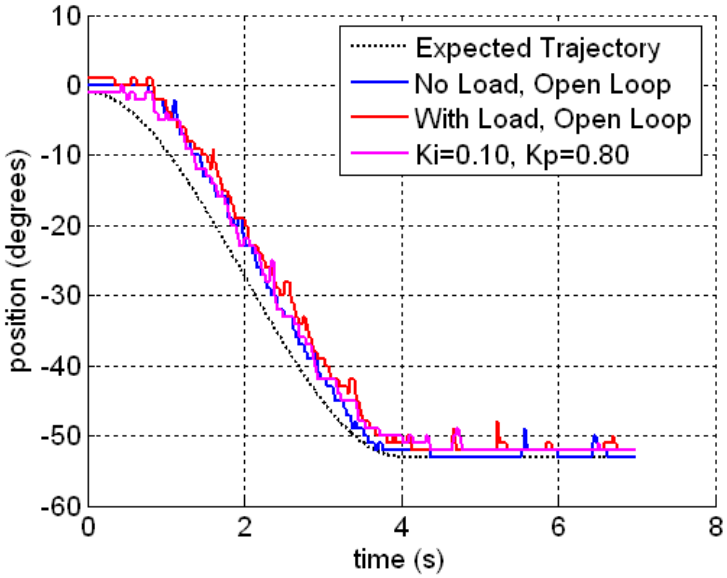

Fig. 107: Junta da anca.

3.6.3. Movimento das duas Pernas

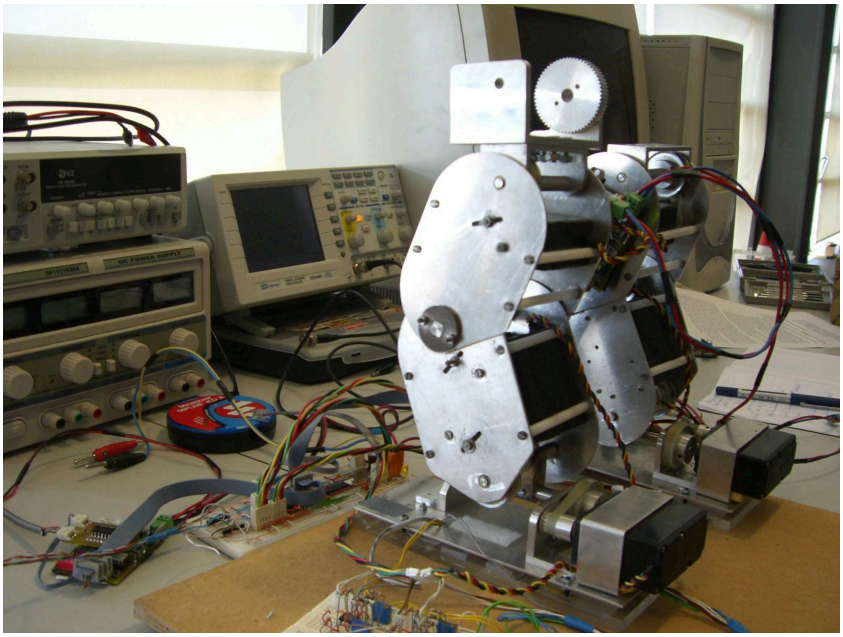

Fig. 108: Movimento de flexão nas duas pernas.

Utilizando agora, as duas pernas juntas (Fig. 108), com a aplicação de uma carga de massa partilhada entre os dois membros, executou-se o mesmo movimento de flexão para teste das juntas do pé, do joelho e da anca. Adicionalmente, de modo a testar a junta lateral do tornozelo do pé, realizou-se um deslocamento para o lado. As figuras seguintes demonstram os resultados referentes a apenas uma das pernas, dado que os da outra perna são bastante semelhantes.

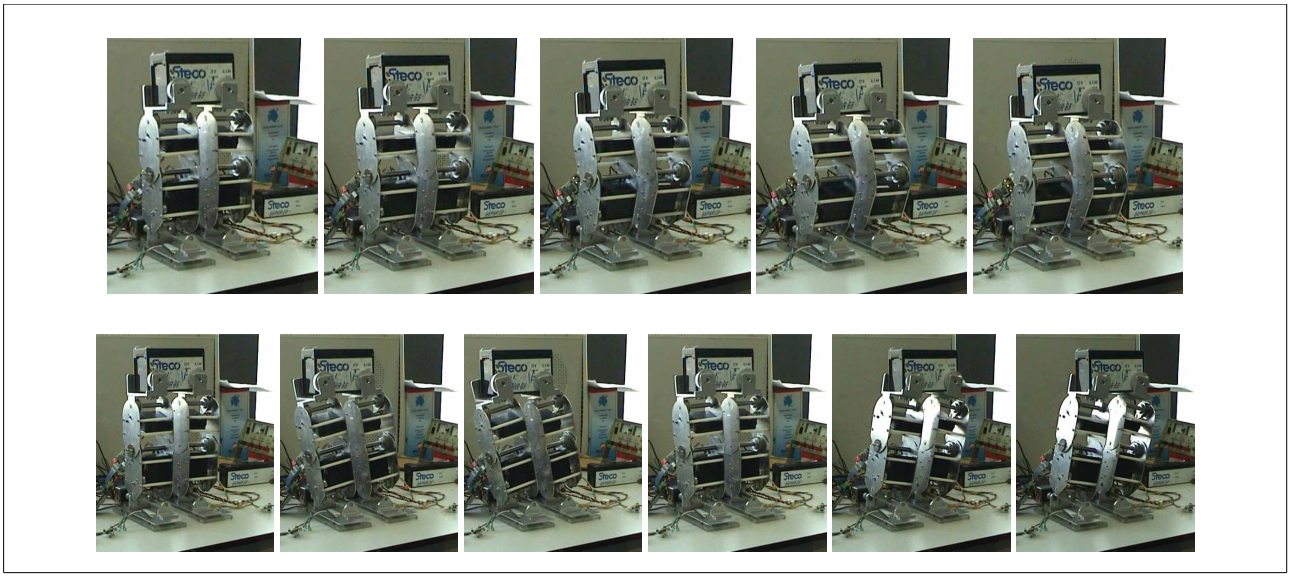

Fig. 109: Conjunto das duas pernas executando um movimento de flexão (sequência superior) e um movimento lateral (sequência inferior) com uma carga de 2.1Kg em completa sincronia de pernas.

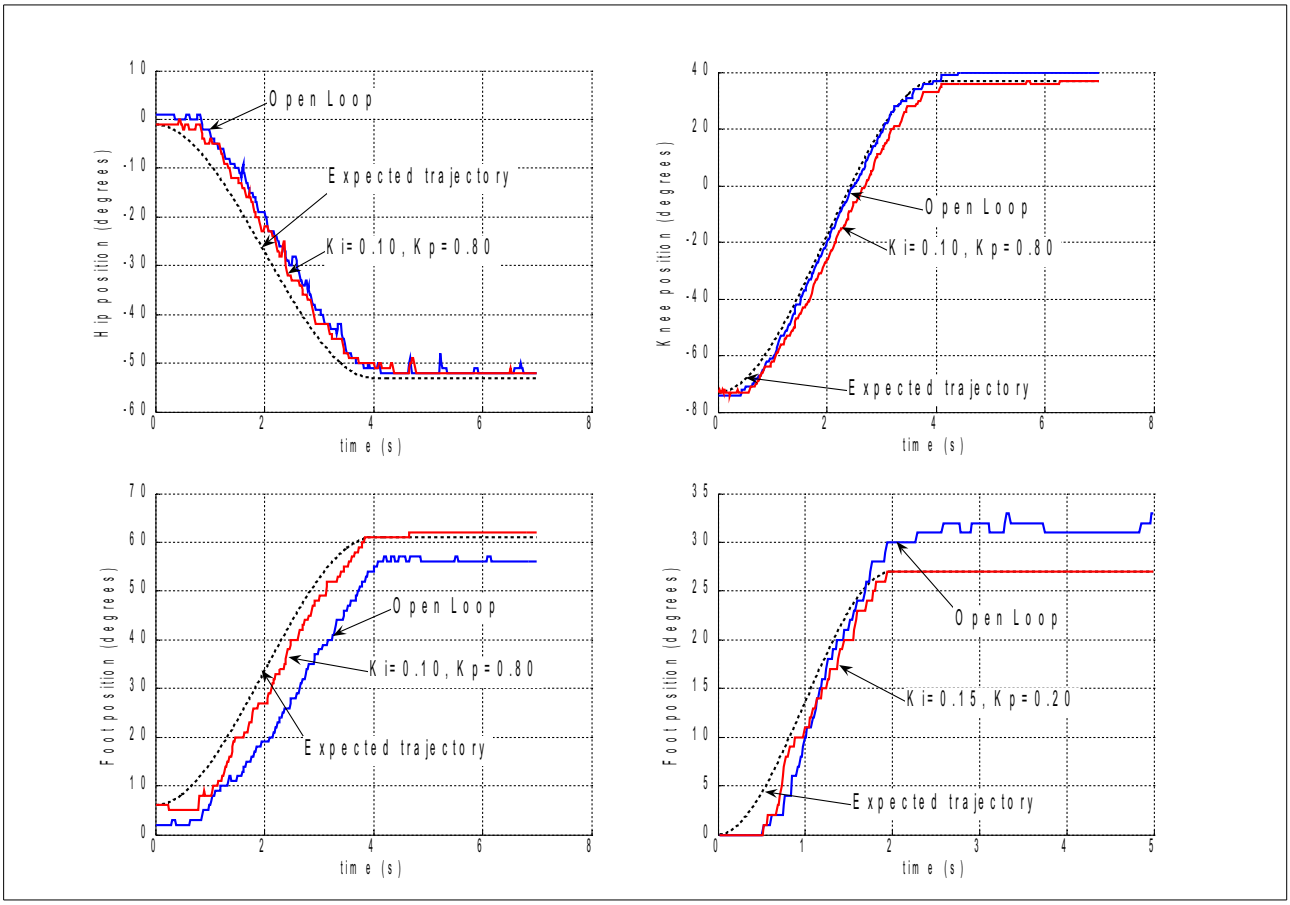

Fig. 110: Resposta ao polinómio com um controlador PI. Imagens superiores e inferior esquerda: comportamento das três juntas envolvidas no movimento de flexão; imagem inferior direita: comportamento da junta lateral do pé na realização do deslocamento lateral.

Como se pode observar, só com a presença do controlador interno, a resposta das juntas mais exigentes (juntas do pé) apresentam um comportamento com um apreciável tempo de atraso e de erro em regime estacionário, que é deveras melhorado com a introdução da compensação PI externa. Demonstra-se assim, com estes dados as vantagens da implementação do controlador.

3.7. ORGANIZAÇÃO DO SOFTWARE

A Fig. 111 visualiza a estrutura do software das unidades de controlo local, já apresentada no capítulo 2 (Fig. 21). Os blocos a verde já foram descritos nesse capítulo e o a vermelho será referido no capítulo 4.

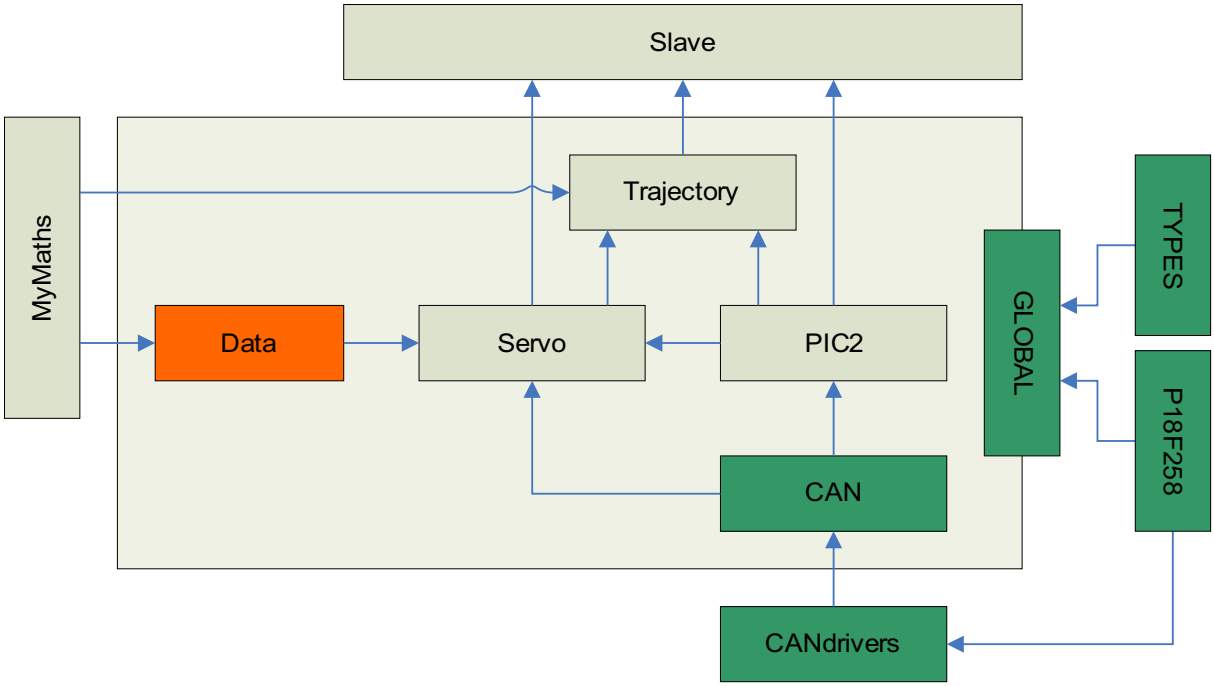

Fig. 111: Relações de inclusão dos módulos de software de cada Slave.

Módulo PIC2

Este módulo define as funções de controlo de baixo nível dos servomotores a serem utilizadas pelas bibliotecas de mais alto nível, e as rotinas de atendimento às interrupções responsáveis pela gestão do PWM de actuação e pela leitura sensorial dos servos. É este módulo que implementa as ideias apresentadas na secção 3.2 sobre o controlo de baixo-nível dos actuadores.

Tabela 32: Funções de acesso externo do módulo *PIC2*.

Função	***Descrição***
initPic	Inicializações relativas às configurações dos periféricos do microcontrolador.
wait	Função bloqueante que gera um atraso de *n* ms (*n* passado como argumento). Esta função faz uso dos *timers* relativos à actuação.
waitTick	Função bloqueante que espera pelo período de PWM seguinte (20ms no pior dos casos).
servoActuation	Actuação directa sobre a posição dos servomotores. As variáveis com a informação do número de iterações a manter o sinal de PWM a 1 durante a zona de descida são actualizadas para as posições solicitadas.
statusPWM	Activação/desactivação dos sinais de PWM à saída do PIC.
statusFilter	Activação/desactivação dos filtros aplicados à posição medida.
limitPosition	Limitação do um valor entre os extremos de posição do servo: -90 e +90º.

A função que mais se destaca desta lista, é sem dúvida, a *servoActuation* pois é ela que define qual deve ser ao *duty-cycle* do sinal de PWM. É esta função que inicializa as variáveis de duração do impulso de PWM que posteriormente serão comparadas com um contador durante a zona de descida de PWM.

Tabela 33: Funções internas do módulo *PIC2*.

Função	***Descrição***
initLocal	Inicialização das estruturas de dados locais ao módulo.
sampleExtraSensors	Leitura dos sensores adicionais (coberto pelo capítulo 4).
updateServoMeasures	Medição iterativa do sinal de saída do servo ao longo de um período de PWM, para cálculo da tensão mínima e da largura do impulso de corrente (Fig. 44).
finalizeServoMeasures	Finalização do processamento sensorial (executado no fim do período de PWM). A posição angular do servo (em graus) e a corrente consumida normalizada entre 0 e 100 são determinados (Fig. 43).
filter	Filtragem da posição medida usando métodos lineares e não-lineares.
delay	Geração de um atraso recorrendo somente à instrução *nop* (*no operation*).
highISR	Rotina de serviço à interrupção de alta prioridade. Nesta rotina é feita gestão da actuação e do processamento sensorial dos servomotores.
lowISR	Rotina de serviço à interrupção de baixa prioridade. Nesta rotina é feita a gestão das comunicações CAN da unidade *slave* (capítulo 2).

A Fig. 112 apresenta um diagrama de blocos do algoritmo em funcionamento na rotina de atendimento às altas interrupções, pondo em prática as ideias apresentadas na secção 3.2.

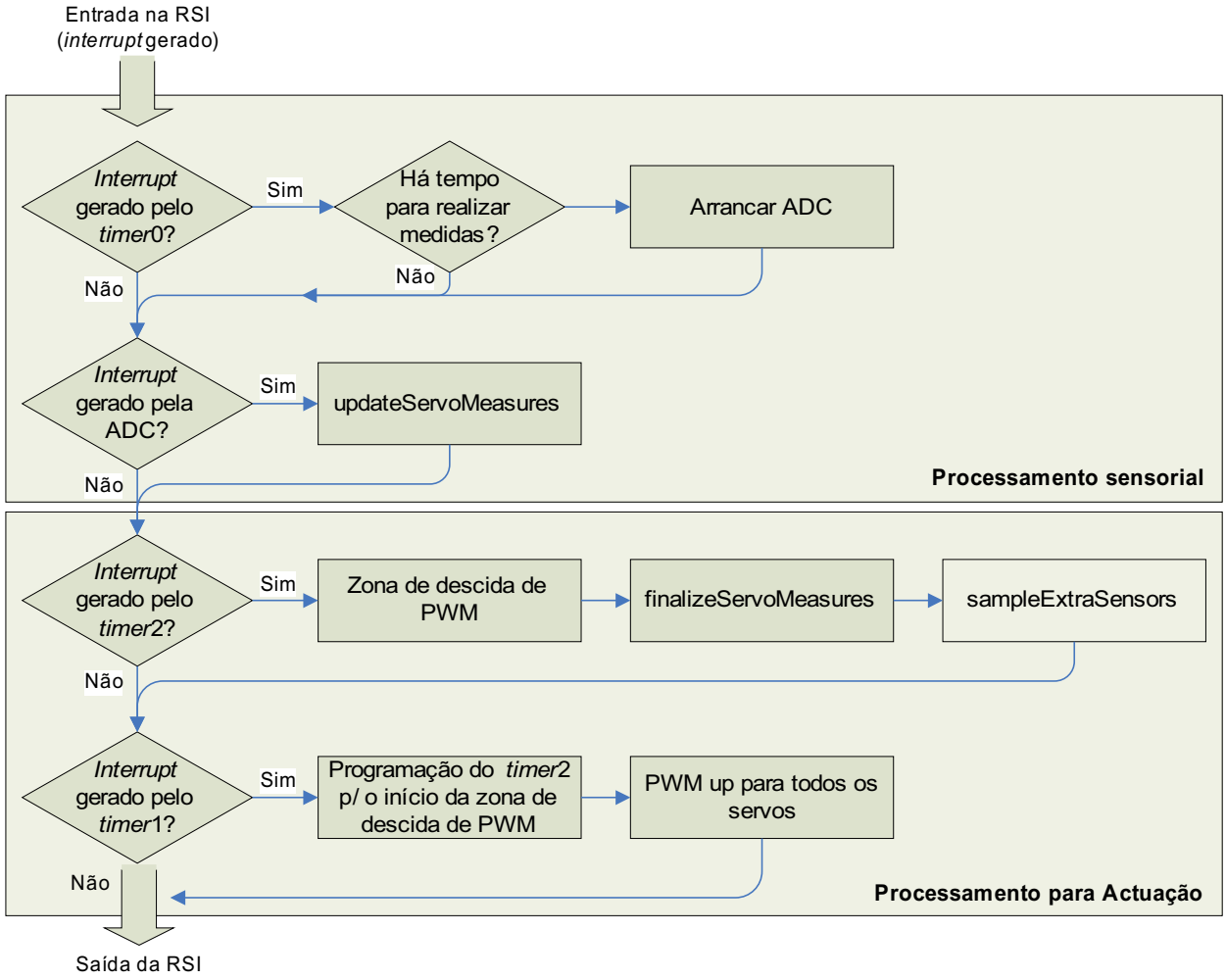

Fig. 112: Algoritmo da RSI de alta prioridade.

Módulo SERVO

Este módulo implementa as rotinas de execução da compensação dos servomotores, uma das quais o controlo de posição segundo o modelo PID descrito na secção 3.4. Outras rotinas podem ser usadas para calibração na medição de posição dos servos ou para definição de parâmetros referência a usar pelos controladores (Tabela 34).

Tabela 34: Funções globais da biblioteca *SERVO*.

Função	*Descrição*
calibration	Calibração dos servomotores. É esperada a primeira ordem de actuação proveniente da unidade Master, e seguidamente é atribuído o PWM correspondente à posição solicitada a cada servomotor. Após o término do deslocamento, a calibração de medição da posição é efectuada com a associação da tensão à saída do potenciómetro à posição solicitada (secção 3.2.3.2).
servoRequest	Atribuição da posição final a atingir. Ao contrário da função *servoActuation* do módulo *PIC2*, a posição solicitada não é atribuída imediatamente ao servo em causa, mas fica armazenada para utilização por parte dos controladores. Estes por sua vez, utilizarão este valor como a posição de referência a ser atingida através dos algoritmos implementados – PID no caso do controlo de posição.
initController	Inicialização das variáveis estáticas utilizadas pelos algoritmos de compensação. Apenas as variáveis associadas ao controlo de posição PID são inicializadas.
controller	Execução dos algoritmos de controlo – um deles é o compensador de posição dos servomotores descrito na secção 3.4.

Esta biblioteca implementa todos os algoritmos de controlo a executar na estrutura humanóide, sendo a compensação PID da posição dos servomotores um deles. A Tabela 4 apresenta os quatro tipos de controlo que podem ser seleccionados através de ordens de actuação executadas pela unidade de controlo principal – o PC – utilizando os barramentos de comunicações RS-232 e CAN. As rotinas que implementam cada um dos quatro algoritmos de controlo apresentam-se na Tabela 35 e são chamadas pela função *controller*.

Tabela 35: Rotinas de implementação da compensação chamadas pela função *controller*.

Função	*Descrição*
openloopControl	Controlo dos servomotores em malha aberta. A posição indicada na chamada da função *servoRequest* é atribuída ao servo em causa, sem passar por qualquer algoritmos de compensação.
locomotionControl	Compensação PID da posição de cada servomotor.
reactionControl	Compensação das forças de reacção aplicadas nos pés segundo o modelo proporcional.
JacobianControl	Compensação das forças de reacção através da matriz Jacobiana.

Os algoritmos de controlo a vermelho na Tabela 35 são utilizados para o controlo de equilíbrio por parte da perna de suporte e serão referenciados com mais detalhe no capítulo 4.

Módulo TRAJECTORY

Este módulo implementa o controlo de velocidade nos servomotores pela aplicação de posições segundo uma determinada trajectória que imprimem a cada servomotor uma determinada velocidade média e eliminam as descontinuidades na forma de deltas de Dirac nas trajectórias de velocidade e de aceleração. A trajectória implementada segue a curva de um polinómio de terceiro grau tal como indicado na Fig. 53. A secção 3.3.2 explica com mais detalhe esta estratégia.

Tabela 36: Funções globais da biblioteca *TRAJECTORY*.

Função	*Descrição*
initTrajectory	Inicialização dos parâmetros estáticos utilizados na realização de cada trajectória.
trajectory	Cálculo e realização iterativa de uma trajectória.

As variáveis indicadas na função *initTrajectory* são apresentadas a seguir:

```
// Estrutura com os dados necessários acerca de um trajecto
typedef struct {
    enum_trajectType type;      // Tipo de trajectória em execução
    double coef[4];             // Coeficientes da função theta=a0+a1*t+a2*t^2+a3*t^3
    word period;                // Duração em ticks da trajectória em execução
    word time;                  // Tempo em curso (em ticks)
    signed char theta_final;    // Posição final da trajectória (valor referência)
    byte control_type;          // Tipo do controlador em aplicação
} struct_trajectory;
static struct_trajectory traject[N_SERVOS];
```

Dos tipos de trajectória que podem ser realizados (*type*) discernem-se dois:

- *Free traject*: Nenhuma trajectória será aplicada, e a posição final *theta_final* a atingir é directamente atribuída ao servomotor em causa. Este tipo de trajecto é aplicado quando a duração solicitada (period) é nula.
- *Normal traject*: Realização da trajectória polinomial de coeficientes *coef*, cuja duração do trajecto é *period* e a posição final é *theta_final*. Esta trajectória é implementada sempre que *period* é positivo.

A Tabela 37 visualiza as rotinas invocadas pelo função global *trajectory*.

Tabela 37: Funções estáticas invocadas pela função *trajectory* da biblioteca *TRAJECTORY*.

Função	*Descrição*
executeTrajectory	Execução de uma de duas trajectórias: *free_traject* ou *normal_traject*.
calcTrajectory	Cálculo de uma nova trajectória a realizar.

A rotina *executeTrajectory* é sempre executada, implementando um dos dois tipos de trajecto apresentados, usando o parâmetro *theta_final* (posição final) no caso *free traject* (trajecto livre), e os parâmetros *time* e *coef* para o caso *normal_traject* (trajecto polinomial):

- *time* indica o tempo decorrido desde o início da trajectória;
- *coef* são os coeficientes do polinómio de terceira ordem $\{a_0, a_1, a_2, a_3\}$ utilizados para o cálculo da posição a aplicar para o instante *time*:

$$\theta = a_0 + a_1 \cdot time + a_2 \cdot time^2 + a_3 \cdot time^3$$

Sempre que a unidade Master solicita uma posição ou um tipo de controlador diferente de *theta_final* e/ou de *control_type* respectivamente, a trajectória em execução é interrompida, e um novo trajecto é calculado tendo em conta os novos valores referência. A rotina *calcTrajectory* realiza esta tarefa calculando uma nova trajectória de acordo com a duração solicitada pelo Master (*period*):

- Duração nula: execução do trajecto livre (*free traject*) indicando a opção através da variável *type*.
- Duração positiva: execução do trajecto polinomial (*normal traject*), calculando para isso os coeficientes *coef* do polinómio de terceiro grau a aplicar.

Em ambos os casos, as restantes variáveis estáticas da estrutura *struct_trajectory* são redefinidas tendo em conta o novo trajecto.

Módulo SLAVE

Programa principal com a inicialização do PIC e a invocação das rotinas de implementação do controlo de posição e velocidade dos servomotores.

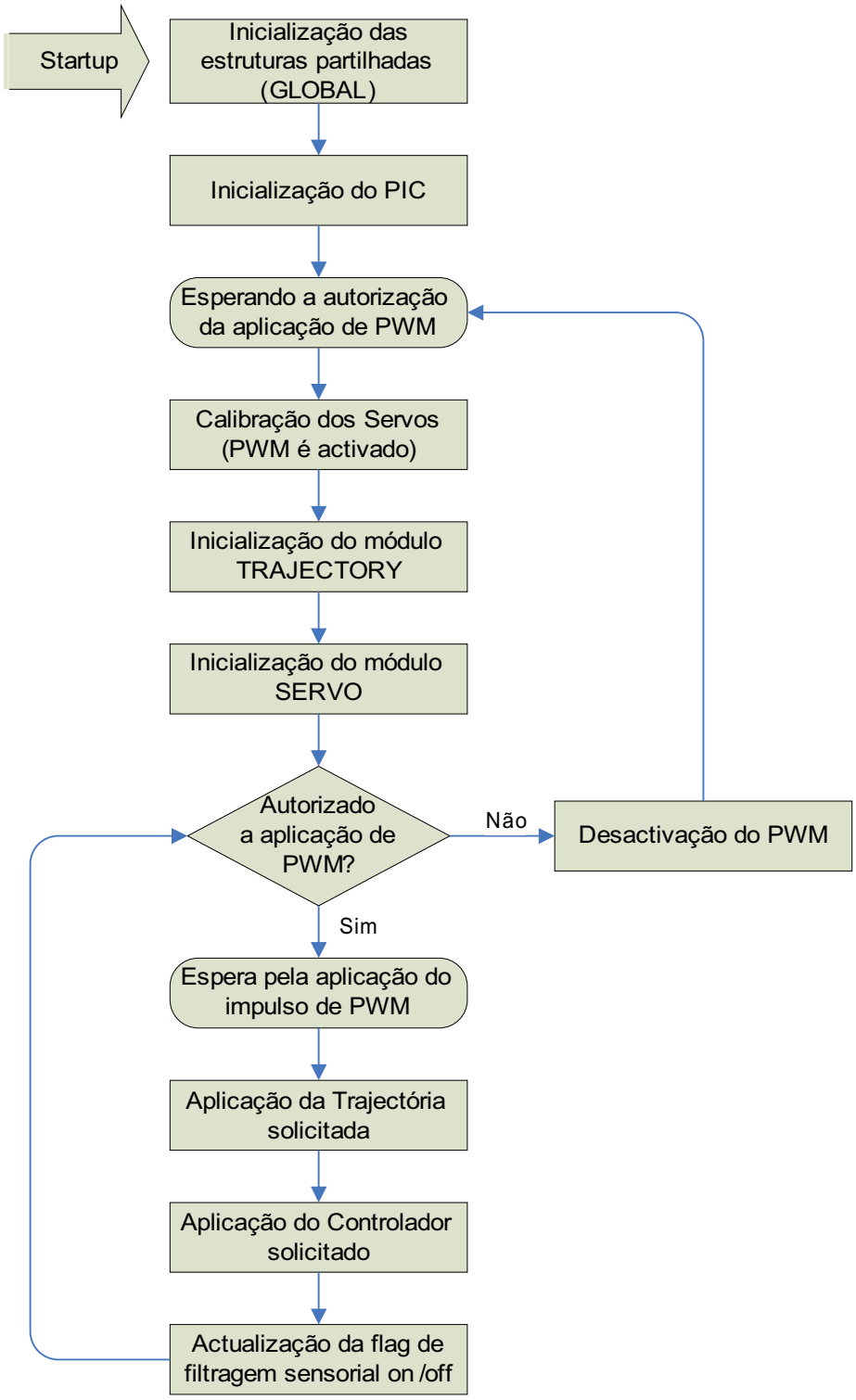

Fig. 113: Algoritmo da função *main* no programa principal *slave*.

Módulo MyMATHS

Esta biblioteca implementa as funções matemáticas necessárias para a programação dos módulos *SERVO* e *TRAJECTORY*. As funções *seno* e *coseno* apenas serão úteis para a implementação do controlador das forças de reacção descrito no capítulo 4.

Função	***Descrição***
abs	Cálculo do módulo de um valor inteiro com sinal *signed int*. Retorno no formato *unsigned int*.
seno	Cálculo da função trigonométrica seno. O argumento é passado em graus e o retorno é um valor inteiro com sinal *signed char* entre -100 e +100: $retorno = resultado \times 100$
coseno	Cálculo da função trigonométrica coseno. O argumento é passado em graus e o retorno é um valor inteiro com sinal *signed char* entre -100 e +100.

As funções seno e coseno são implementadas recorrendo a uma *lookup table* com a associação do resultado a cada valor angular (Tabela 38).

Tabela 38: *Lookup table* para a função seno.

Ângulo **(º)**	***Seno*** **(×100)**
0	0
1	1
2	3
3	5
4	6
5	8
...	...
81	98
82	99
83	99
...	...
89	99
90	100

Apenas nos é útil armazenar a gama de 0 a +90º, uma vez que o conjunto de resultados repete-se para outros ângulos fora deste intervalo. Deve-se, contudo, adaptar o argumento para a gama [0,90]º e aplicar correctamente o sinal ao resultado final.
No que respeita à resolução da *lookup table*, é suficiente armazenar o resultado para cada ângulo inteiro, uma vez que o algoritmos de medição sensorial do potenciómetro de posição de cada servomotor também só consegue medir com uma resolução de 1º.

De notar que a *lookup table* da função seno (Tabela 38) também pode ser utilizada para o cálculo do coseno dado que:

$$coseno(\theta) = seno(90^{\circ} - \theta)$$

... pelo que apenas uma *lookup table* é suficiente.

3.8. RESOLUÇÃO DE ANOMALIAS

Embora, em função dos dados sensoriais, as respostas possuem uma boa qualidade, a nível mecânico já não é bem assim, com os efeitos elásticos das correias de transmissão e as folgas das juntas a tornaram-se significativos. Ao fim de cada conjunto de testes com uma carga elevada, verificava-se um deslocamento do parafuso de aperto de cada correia resultando no aparecimento de uma folga nas juntas mais exigentes. Tal exigia o reajuste do parafuso de aperto para corrigir a folga (Fig. 114).

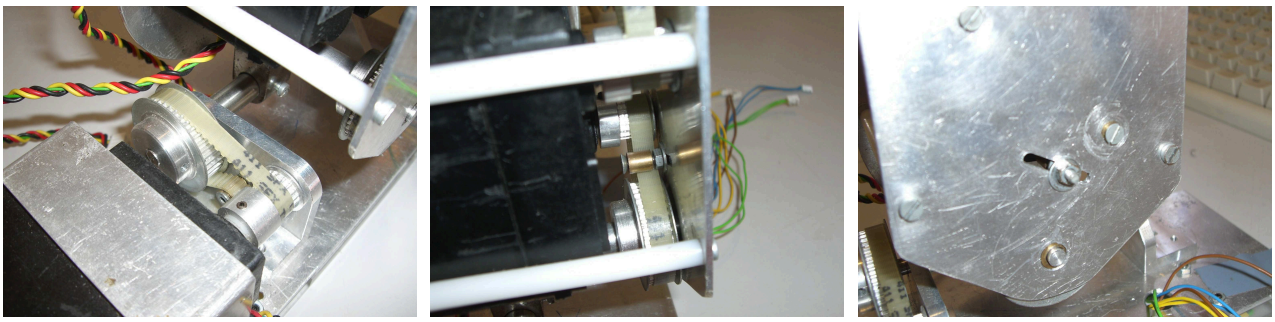

Fig. 114: Aperto das correias de transmissão: junta lateral (esquerda) e dianteira do pé (restantes).

Um outro problema mais complexo prende-se com a ligação da junta do joelho com uma chaveta que garante a ligação da perna inferior com a superior. Ao fim de algum tempo começou a ocorrer folga nesta ligação, resultando numa degradação permanente dos movimentos, dado que esta folga não é reajustável. Recomenda-se por isso, a substituição do material utilizado por outro mais resistente, ou na impossibilidade desta solução, adicionar mais uma chaveta de fixação no outro lado do joelho (Fig. 115 à direita).
Também se verificaram algumas folgas permanentes ao nível do veio do tornozelo com a presença de algumas ligações deficientes entre este veio e o veio do pé. Recomenda-se, também a revisão destas ligações (imagem esquerda e central da Fig. 115).

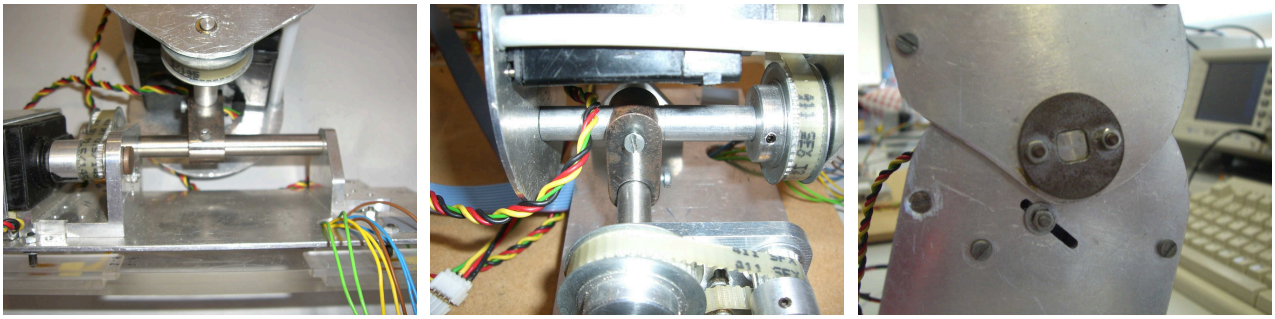

Fig. 115: Localização das principais folgas existentes: juntas lateral (esquerda)e dianteira (central) do pé e chaveta de ligação do joelho (direita).

Como se pode concluir, muito embora o comportamento dos servomotores seja considerada como muito bom pelos resultados sensoriais, em termos mecânicos ainda há muitos aspectos que precisam de ser melhorados sob pena de tornar ineficaz os algoritmos de controlo de posição e velocidade. Aconselha-se, por isso, uma séria revisão à estrutura das pernas, uma vez que são estes membros que terão de suportar os esforços mais exigentes, nomeadamente:

- ➔ no aperto das correias de transmissão: estas são as principais causas de folgas, aumentando à medida que se verifica um esforço, correndo o risco de provocar saltos de dentes. Uma solução que já foi indicada corresponde a efectuar o aperto de forma dinâmica com o recurso a uma espécie de mola, tornando automático o processo de ajuste.
- ➔ ao nível das ligações entre os veios e os elos tornando-os mais robustos quer pelo aumento da resistência dos materiais quer pelo aumento do número de ligações. Tal convêm ser estudado;

3.9. CONCLUSÕES

Este capítulo teve como objecto de estudo, os servomotores utilizados para actuação. Como são dispositivos relativamente pequenos e baratos que já oferecem o controlo de posição, foram escolhidos dada a sua simplicidade de interface com um microcontrolador – podem-se ligar directamente – podendo controlar a sua posição simplesmente através de um sinal digital de PWM cuja largura de pulso define a posição do servo. Os modelos da HITEC, nomeadamente o modelo HS-805BB para as juntas de grande esforço, foram escolhidos dada o seu elevado binário (2.42 N.m @6V) comparativamente a outras marcas.

No entanto, cedo nos apercebemos que estes dispositivos não são perfeitos não oferendo qualquer tipo de controlo de velocidade – simplesmente se deslocam para a posição final à sua velocidade máxima – algo nada favorável quando pretendemos controlar a posição e a velocidade das juntas para que realizem movimentos suaves de velocidade limitada. Contudo, pela aplicação de um conjunto crescente (ou decrescente) de pequenos degraus de posição em forma de rampa, conseguimos implementar uma espécie de controlo de velocidade indirecto, na medida em que os extremos de posição da rampa e o intervalo de tempo da sua duração impõe uma velocidade média do movimento. Uma forma de trajectória mais exigente é o polinómio de terceiro grau que além de permitir a definição de uma velocidade média garante outros aspectos como é o caso de velocidade nula e acelerações limitadas no início e no fim da trajectória, algo importante para prevenir eventuais picos de corrente e limitar o consumo de corrente durante o percurso na realização destas trajectórias.

Mais tarde verificou-se que embora estes actuadores apresentem respostas praticamente ideais (resposta semelhante à trajectória solicitada), na presença de cargas tal deixa de ser verídico verificando um acentuado tempo de atraso no acompanhamento da trajectória solicitada e uma dificuldade em atingir o valor final devido à presença da força gravítica no binário aplicado (erro em regime estacionário não nulo). Usando o sinal de *feedback* de posição que o servo disponibiliza, por comparação com a posição desejada, tentou-se implementar uma lei de controlo que permitisse corrigir estes desvios. Esta lei de controlo baseou-se num compensador clássico do tipo PID (Proporcional+Integrador+Derivador), que por meio de diversos ensaios, conseguimos associar cada componente às características da resposta de modo a permitir-nos escrever um procedimento para rapidamente encontrarmos, através de ensaios, um conjunto de parâmetros que aproximem bastante a resposta do servo à desejada.

No entanto um conjunto de parâmetros do controlador só se adequa a uma situação específica de carga, trajectória e velocidade específicos, pelo que seria importante detectar os vários cenários possíveis para que o controlador se adaptasse. Tal seria possível pela medição do binário aplicado no motor que varia de acordo com a inércia presente.
Usando o sinal de saída do potenciómetro interno do servo, conhecido por estar relacionado com a sua posição, descobrimos que além de nos fornecer a posição também nos providencia a corrente corrente consumida pelo dispositivo, que por sua vez está relacionada com o binário aplicado ao motor. Desta forma poderíamos aplicar um controlador de força que adaptaria os parâmetros de controlo de acordo com a corrente drenada pelo motor. No entanto este sinal apresenta uma natureza bastante oscilatória, ainda mais com o controlo de posição, estando relacionada não só com o binário resultante da força gravítica como também do que resulta da velocidade e da aceleração. Para já desconhece-se qual seria o formato da lei de controlo deste adaptador, nem que parâmetros deverá assumir pela que mais investigação será necessária sobre este tópico.

Mesmo assim, nas experiências que temos realizado sobre as pernas do humanóide temos vindo a utilizar parâmetros fixos do controlador de posição, que claro, são bastante limitados para garantir a estabilidade em qualquer uma das juntas devendo, estes valores, corresponder ao pior caso. Infelizmente ficamos a perder, que em grande das juntas os parâmetros não são os mais optimizados introduzindo atrasos adicionais que por vezes podem ser piores do que se o controlador estivesse desligado (em malha aberta).
Efectuando ensaios sobre as pernas acabámos por verificar que as exigências em termos de binário são bastante inferiores às testadas com o servo isolado, com respostas de bastante qualidade mesmo em malha aberta. Apenas a estrutura mecânica das pernas se revelou incapaz de executar movimento com as cargas mais pesadas provocando folgas, que embora algumas fossem ajustáveis, outras revelaram-se sem solução com a necessidade de substituir peças. Esperemos que no trabalho futuro que se segue, estas contrariedades

sejam levadas a sério não só no sentido de eliminar quaisquer folgas existentes, como também em aplicar novas técnicas e materiais que atribuam robustez à estrutura, independentemente da solicitação imposta.

Outros aspectos que precisam de ser limados prendem-se com a alimentação dos actuadores e das unidades de controlo local. Mesmo com o uso de duas baterias de elevado fornecimento de corrente em paralelo, verifica-se esporadicamente um aparente *reset* de alguns PIC's, na medida que por vezes alguns servos deslocam-se para a posição original a alta velocidade. Como esta anomalia é muito esporádica iremos por de parte bugs do software.

- Considerando a hipótese de um *reset* por parte de um *slave*, nada deveria acontecer, uma vez que quando iniciam o seu funcionamento esperam por um comando do *master* e só depois aplicam o PWM com a posição válida, que deverá corresponder à posição actual. Se alguma coisa ocorrer é um deslocamento mínimo devido ao controlador PID. Por isso, este não deverá ser o problema.
- Se o *reset* ocorresse no *master*, todos os *slaves* dirigir-se-iam para as suas posições originais e não apenas um ou outro como se observa, pela que esta opção também é posta de parte.

Logo aparentemente, o problema não se deve a um *reset*. Eventualmente pode ser devido a informação incorrecta nas mensagens CAN. Apenas é estranho dado que as mensagens de actuação teriam de possuir valores 0 nos campos de posição e velocidade durante imenso tempo para serem visualizados os efeitos. Se se deve a isto, em princípio resolvendo bug do bloqueio do barramento CAN enunciado nas conclusões do capítulo 1, deveremos também resolver este problema.